May 3, 2024

Group Theory for Physicists

Kasper Peeters

These notes are incomplete, unfinished and most likely contain errors. They are solely meant as a guide to complement your notes taken during the lectures and cover a lot of the material discussed there, but certainly not everything.

Lecture notes for the "Group Theory" module of the M.Sc. in Particles, Strings and Cosmology at Durham University.

Department of Mathematical Sciences
University of Durham
South Road
Durham DH1 3LE
United Kingdom

kasper.peeters@durham.ac.uk

1 Introduction

1.1. Why group theory?

Group theory is, in short, the mathematics of *symmetries*. You already know that symmetries can be very important in understanding or simplifying physics problems. When you study classical mechanics, you learn that symmetries of a system are intimately related to the existence of conserved charges. Their existence often makes solving for the dynamics a lot simpler. Even if a symmetry is not present exactly (for instance, when a system is almost-but-not-quite spherically symmetric), we can often describe the system as a small perturbation of a system that does exhibit symmetry. A surprisingly large number of physics problems is built around that idea; in fact, practically all systems for which we can solve the dynamics exactly exhibit some sort of symmetry that allow us to reduce the often horrible second-order equations of motion to much simpler first-order conservation equations.

Dynamics (governed by 2^{nd} order equations) simplifies tremendously in the presence of conserved charges (involving only 1^{st} order derivatives, typically).

Symmetries have three properties which are key in understanding their mathematical structure: they are associative, there is an identity element, and for every symmetry transformation there is an inverse transformation that cancels it and brings us back to the original system. These three properties are what defines a "group", and the theory of these mathematical structures is "group theory". The goal of this module is then, simply put, to show you *which types of symmetries there are* (the "classification" of groups) and *how they can be made to work in concrete physical systems* (how their "representation" on physical systems works).

1.2. A quick tour

1.2.1 Definitions and basic concepts

Translating the basic properties of a symmetry transformation, mentioned above, into a mathematical language leads us to the definition of a *group* G:

Properties of a group

1. For two elements $x, y \in G$ the composition $x \cdot y \in G$, too (a property called "closure").
2. There exists an identity element, denoted e, such that $e \cdot x = x \cdot e = x$ for every $x \in G$.
3. For every $x \in G$ there exists an inverse x^{-1} such that $x \cdot x^{-1} = x^{-1} \cdot x = e$.
4. The composition is associative, that is, $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ for any $x, y, z \in G$.

This sounds like a bunch of trivialities, but there certainly are counter-examples. For instance, the set $\mathbb{R}$ of real numbers with multiplication as the composition operation

does *not* form a group: the element 0 does not have an inverse.

There are two basic classifications of groups which we can already discuss. First, note that generically, a group does not need to be commutative, that is, $x \cdot y \neq y \cdot x$ in general (matrix multiplication for instance, typically does not satisfy this, except for special matrices). If commutativity does hold for all elements, then we have a so-called *Abelian group*.

The other classification is based on whether we have a discrete number of elements in the group, or a continuum of them. Discrete groups, for instance, include the permutation group S_n of n elements. Continuous groups include the rotation group $SO(3)$ of points in three dimensions. Groups which depend continuously on one or more parameters are also called *Lie groups*. In this module, we will mostly be interested in the latter. The classification of discrete groups follows quite a different set of steps than the classification of Lie groups, for reasons that will become clear shortly.

The most well known Lie groups are what are known as the *classical matrix groups*. These can, as the name suggests, be defined in terms of generators which are given by matrices, with some additional constraints which occur often in classical or quantum mechanical properties. The full list is:

List of classical groups

$GL(N, \mathbb{R})$
: The general linear group in N dimensions.
Given by all invertible $N \times N$ matrices with elements in $\mathbb{R}$. A similar story holds for matrices with elements in $\mathbb{C}$.

$SL(N, \mathbb{R})$
: The special linear group.
Those matrices of $GL(n, \mathbb{R})$ which have $\det A = 1$.

$O(N)$
: Orthogonal group,
All matrices satisfying $AA^T = 1$.

$SO(N)$
: Special orthogonal group, again adding $\det A = 1$ as a condition to the orthogonal group.

$U(N)$
: Unitary group, for which $A^\dagger A = 1$ where $A^\dagger := (A^*)^T$.

$SU(N)$
: Special unitary group, those unitary matrices which also satisfy $\det A = 1$.

$Sp(2N)$
: All matrices that satisfy $\Omega A + A^T \Omega = 0$, with

$$\Omega = \begin{pmatrix} 0 & \mathbb{1}_N \\ -\mathbb{1}_N & 0 \end{pmatrix} .$$

There are various other matrix Lie groups not covered by this list, but those have more complicated matrix conditions not easily guessed from some physics problem. We will encounter them when we turn to the classification problem.

In order to illustrate some more of the concepts of Lie groups, and to connect them to what are known as Lie algebras, let us focus on the rotation group in 3 dimensions (or $SO(3)$ in the language introduced above) and discuss a few of its

elements. Rotations around the z-axis are obtained by acting on points with the one-parameter matrix

$$R_z(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \approx 1 - i\delta\phi T_z \tag{1.1}$$

where

$$T_z = \begin{pmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} . \tag{1.2}$$

In fact, the relation between $R_z(\phi)$ and T_z can be given to all orders, not just the first order, and reads

$$e^{-i\phi T_z} = R_z(\phi) . \tag{1.3}$$

The matrix T_z is called an element of the *Lie algebra* corresponding to the Lie group of which $R_z(\phi)$ is an element. Unsurprisingly, the group formed from just $R_z(\phi)$ is called a *subgroup* of $SO(3)$, in fact, these matrices form $SO(2)$.

In order to compute the product of Lie group elements from their definition in terms of Lie algebra generators, we in fact only need to know the commutators of those generators. This is because the Baker-Campbell-Hausdorff expressions, which give the product of matrix exponentials, only involve commutators on the right hand side. For example, if $[X, Y] = \mathbb{1} \cdot \text{const.}$, we have

$$e^X e^Y = e^{X+Y+\frac{1}{2}[X,Y]} . \tag{1.4}$$

The general case is more complicated, but never requires knowledge of the product of two Lie algebra matrices, only knowledge of the commutator. Therefore, it makes sense to consider the abstract mathematical structure of a *Lie algebra*, which is formed by elements X_a together with a Lie bracket,

$$[X_a, X_b] = i f_{abc} X_c . \tag{1.5}$$

The f_{abc} are called *structure constants*. For matrix groups these X_a are of course also matrices, and the Lie bracket is a commutator. However, the important thing is that we never really need to know the product of matrices X_a, only their commutator, so it makes more sense to view the Lie bracket as a single operation, rather than one built from a product and a difference. The Lie product should furthermore satisfy the condition that it is linear in its arguments, that it is anti-symmetric, and that it satisfies the Jacobi identity,

$$[X_a, [X_b, X_c]] + [X_c, [X_a, X_b]] + [X_b, [X_c, X_a]] = 0 . \tag{1.6}$$

Again, these are automatic for matrix algebras, but can be viewed as a more general definition of a mathematical structure.

We will here only consider Lie algebras with a finite number of elements. There also exist Lie algebras with an infinite number of elements, of which the most commonly encountered ones in physics are the Kac-Moody algebras.. They occur in the study of two-dimensional conformal field theories.

Kac-Moody algebras

One finally also encounters non-matrix Lie algebra, that is, algebras which are not commonly expressed or defined in terms of matrix generators. An example is the Heisenberg algebra,

$$[X, Y] = [X, Z] = 0 , \quad [Y, Z] = X . \tag{1.7}$$

The standard representation is in terms of differential operators acting on functions,

$$Y = \frac{\mathrm{d}}{\mathrm{d}t} , \quad Z = t , \quad X = 1 . \tag{1.8}$$

We will not have much to say about such representations either.

1.2.2 Observing symmetries in quantum mechanics

We will see later that there are also strong consequences of symmetry in quantum field theory, but let us stick to quantum mechanics for the time being. In quantum mechanics, the fact that a system is invariant under a particular symmetry transformation means that the Hamiltonian remains unchanged if we perform that transformation. In a formula, we have

$$[\hat{H}, \hat{A}] = 0\,, \tag{1.9}$$

where $\hat{H}$ is the Hamiltonian and $\hat{A}$ is one of the generators of the symmetry group. Let us try to see what this implies for the time evolution of states. States are labelled by eigenvalues of the conserved charges. If we consider an $SO(3)$ system as example, this means that we label states by two eigenvalues, of $\hat{J}^2$ and $\hat{J}_z$ for instance. Call these l and m. Now in general, the time evolution of a state is given by

$$|\psi(t)\rangle = e^{i\hat{H}t}|\psi(0)\rangle\,, \tag{1.10}$$

so what appears in expressions for transition amplitudes are matrix elements of the Hamiltonian. The symmetry condition (1.9) puts a strong constraint on these matrix elements, as can be seen from the following expression

$$0 = \langle l', m'|[\hat{L}_z, \hat{H}]|l, m\rangle = (m' - m)\langle l', m'|H|l, m\rangle\,. \tag{1.11}$$

The matrix element can only be non-zero when $m = m'$. This is a so-called *selection rule*: there can be no transition from a state with $m \neq m'$ in systems which are rotationally invariant.[1]

Another way in which the consequences of symmetry can be measured is by noting that if a state transforms as a vector under some symmetry group, it necessarily comes in a multiplet together with other states carrying the same energy. If you now perturb the system slightly, in such a way that you break the symmetry, you can observe *level splitting*.

(end of lecture 1)[2]

1.2.3 Abstract groups/algebras versus representations

One of the main reason for being a bit pedantic about defining groups and algebras in an abstract sense is that there is almost always more than one way to realise the same given group or algebra in a physical system. This leads us to define a *representation* of a group, of dimension p, as a map from the group elements to a set of $p \times p$ matrices $D(X)$ acting on a p-dimensional vector space, which preserve the group composition law,

$$D(X_1 \cdot X_2) = D(X_1)D(X_2)\,, \quad \text{for all } X_1, X_2 \in G\,. \tag{1.12}$$

The product on the right denotes ordinary matrix multiplication. For a representation of an algebra, we would require

$$D([x_1, x_2]) = [D(x_1), D(x_2)]\,, \quad \text{for all } x_1, x_2 \in g\,. \tag{1.13}$$

Let us have a look at some examples of different representations. Consider the Lie algebra $sl(2, \mathbb{C})$, which is formed by all trace-less 2×2 complex matrices. Since we consider the algebra over $\mathbb{C}$, these matrices can be built using 3 generators; all other matrices are then obtained by taking suitable complex linear combinations. A useful basis is[3]

$$T_1^{(f)} = \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix}\,, \quad T_2^{(f)} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\,, \quad T_3^{(f)} = \begin{pmatrix} i & 0 \\ 0 & -i \end{pmatrix}\,. \tag{1.14}$$

[1] A refinement of this argument goes under the name of the Wigner-Eckart theorem, to be discussed later.

[3] These anti-hermitean matrices are related to the perhaps more familiar Pauli matrices by multiplication with plus or minus i.

These matrices satisfy the algebra

$$[T_i^{(f)}, T_j^{(f)}] = 2\epsilon_{ijk} T_k^{(f)} \,. \tag{1.15}$$

They form the so-called *fundamental* or *defining* representation of the algebra $sl(2)$. However, the $T_i^{(f)}$ are certainly not the only matrices that satisfy these commutation relations. It is straightforward to verify that the same commutators are obtained using the following 3×3 matrices,

$$T_1^{(a)} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -2 \\ 0 & 2 & 0 \end{pmatrix} , \quad T_2^{(a)} = \begin{pmatrix} 0 & 0 & 2 \\ 0 & 0 & 0 \\ -2 & 0 & 0 \end{pmatrix} , \quad T_3^{(a)} = \begin{pmatrix} 0 & -2 & 0 \\ 2 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} . \tag{1.16}$$

This representation is called the *adjoint* representation, for reasons that will become clear in the next chapter. In any case, the existence of these matrices shows that the same algebra can have different representations.

Fundamental representation, conjugate fundamental representation, trivial representation.

Tensor products, show first by writing out, then do in index notation following Mukund. Discuss reducibility -¿ block diagonal. Discuss complete reducibility and Maschke's theorem.

$$\mathbf{2} \otimes \mathbf{2} = \mathbf{3} \oplus \mathbf{1} \,. \tag{1.17}$$

Some Lie algebras do not admit representations which are finite-dimensional (i.e. the generators cannot be written down as matrices with a finite number of rows/columns). Such Lie algebras are called non-matrix Lie algebras, and their existence shows that it is important to consider Lie algebras as mathematical structures by themselves, not necessarily defined in terms of matrices. Do not confuse these with the infinite-dimensional algebras we mentioned earlier.

Non-matrix Lie algebras

1.3. Literature and software

There are plenty of unreadable books on group theory that focus on all the gory mathematical details without ever discussing a physics example. A few books not in this category are

- .
  A nice and compact book written for physicists.
- .
  More thorough and mathematical book, with lots of examples and exercises, written from the perspective of mathematical physics.
- .
  Another very compact book, perhaps a bit too compact. Now also available for free online,

  http://phyweb.lbl.gov/~rncahn/www/liealgebras/book.html

A few others which I personally do not like too much, but some people swear by these:

- .
  More physics than some of the other books on this list. Has a nice discussion of Young tableaux and of the Lorentz group.

- .
  More mathematically oriented book, now out of print, but available in online from

  http://www.math.cornell.edu/~hatcher/Other/Samelson-LieAlg.pdf

- .
  Much more mathematically oriented. Starts (as the title suggest) right from representation theory.

When we get a bit further on in the module, you might be interested to play with Lie algebras yourself. Rather than doing it by hand, which can be cumbersome, there is some good software out there which can help you build a better understanding:

- LiE, a computer algebra package for Lie group computations, M.A.A. van Leeuwen, A.M. Cohen and B. Lisser.
  Can be used to construct weight systems, to find branching rules, and many many other things which we do in this course. Available under the LGPL license from

  http://www-math.univ-poitiers.fr/~maavl/LiE/

  On Debian/Ubuntu Linux systems, install the `lie` package. Examples using LiE are included at various points in these notes, in the form of small bits of code in the margin labelled by "LiE".

- Cadabra, a field-theory motivated approach to computer algebra, K. Peeters.
  Contains various algorithms to deal with Young tableaux. Available from

  http://cadabra.phi-sci.com

  On Debian/Ubuntu Linux systems, install the `cadabra` package. Examples using LiE are included at various points in these notes, in the form of small bits of code in the margin labelled by "Cadabra".

2
Lie algebra classification

We now turn to the key question, namely to find a procedure that provides for us a list of all possible Lie algebras. The key ingredient in this classification is to choose a convenient basis of the generators, and then use this basis to show that all Lie algebras can be reduced to combinations of $sl(2)$ algebras, connected to each other by specific non-zero commutation relations between generators of these $sl(2)$s. All that is needed to work this out is knowledge about the representation theory of $sl(2)$. To give an idea of how this procedure works, we will first discuss $sl(3)$ in this language. We will then turn to the Cartan classification using Dynkin diagrams.

2.1. Main ingredients

2.1.1 The adjoint representation

We have seen several examples of representations already, but there is one representation that needs special attention because it is crucial in the present chapter. This representation is called the *adjoint* representation, and is obtained by thinking of the generators themselves as vectors. There is a natural action of the algebra elements on themselves, given by the action of the commutator.

$$\text{ad}\, x(y) = [x, y]\,, \quad \text{also written as} \quad \text{ad}\, x = [x, \cdot]\,. \tag{2.1}$$

This preserves the algebra structure, as is readily shown by making use of the Jacobi identity,

$$\begin{aligned} [\text{ad}\, x, \text{ad}\, y]w &= [x,[y,w]] - [y,[x,w]] \\ &= [x,[y,w] + [y,[w,z]] \\ &= -[w,[x,y]] \\ &= [[x,y],w] = \text{ad}[x,y](w)\,. \end{aligned} \tag{2.2}$$

The adjoint representation clearly always exists, and has a dimension equal to the number of generators of the algebra. If required you can of course write it in more usual matrix-vector multiplication form (like we did earlier for the two-tensor product representation).

2.1.2 Representation theory of $sl(2)$

The second crucial ingredient for this chapter is the representation theory of the simplest of Lie algebras, namely $sl(2)$. Let us first rewrite the $sl(2)$ algebra in a basis

which is more useful for this chapter. Instead of using the $T_i^{(f)}$ matrices, we will use the 'cleaner' ones

$$H = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad E_+ = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad E_- = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}. \tag{2.3}$$

These are simple complex linear combinations of the $T_i^{(f)}$, so they generate the same complex algebra. Their commutation relations read

$$[H, E_\pm] = \pm 2 E_\pm\,, \quad [E_+, E_-] = H\,. \tag{2.4}$$

It is these commutation relations that we want to use to study $sl(2)$.

So let us now think of the $H, E_\pm$ as abstract generators and study the possible ways in which we can realise them. The starting point will be that we will choose a basis of the representation space in which H^1 is diagonal, i.e. in which

$$H v_\lambda = \lambda\, v_\lambda\,. \tag{2.5}$$

The $E_\pm$ operators are 'ladder operators', in the sense that they raise or lower this eigenvalue,

$$H(E_\pm v_\lambda) = E_\pm H v_\lambda \pm 2 E_\pm v_\lambda = (\lambda \pm 2)\, E_\pm v_\lambda\,. \tag{2.6}$$

Therefore, if you start with some v_λ, you can construct all the vectors in that representation by laddering. Their H eigenvalues go up or down by two every time you act.

We will assume now that the representation is finite-dimensional (this can be shown to be true by virtue of the fact that the generators are hermitian, but we will not waste time with that). Since the dimension is finite, there has to be a vector which has the highest H^1 eigenvalue, which we will call Λ.

Representations of $sl(2)$ are labelled by Λ, the *highest weight* (largest eigenvalue of H).

$$v_{\Lambda-2n} := (E_-)^n v_\Lambda\,. \tag{2.7}$$

After a finite number of steps, the string has to break,

$$E_- v_{\Lambda-2N} = 0\,. \tag{2.8}$$

To get more information, we have to look at how the raising operator acts. We get

$$E_+ v_{\Lambda-2} = E_+ E_- v_\Lambda = [E_+, E_-] v_\Lambda = H v_\Lambda = \Lambda\, v_\Lambda\,. \tag{2.9}$$

We get back a scalar multiple of v_Λ (important: you might have guessed, based on (2.7), that this number is one, but it is not). For a generic starting point, the scalar multiple is also easily computed; defining

$$E_+ v_{\Lambda-2n} = r_n\, v_{\lambda-2n+2}\,, \tag{2.10}$$

we get

$$E_+ v_{\Lambda-2n} = E_+ E_- v_{\Lambda-2n+2} = (E_- E_+ + H) v_{\Lambda-2n+2} = (r_{n-1} + \Lambda - 2n + 2)\, v_{\Lambda-2n+2}\,. \tag{2.11}$$

We thus have the recursion relation

$$r_n = r_{n-1} + \Lambda - 2n + 2\,, \tag{2.12}$$

with $r_0 = 0$ because the chain breaks at the highest eigenvalue. This recursion relation has the solution

$$r_n = n(\Lambda - n + 1)\,. \tag{2.13}$$

Just like $r_0 = 0$, we also need the chain to break off at the lower end,

$$0 = E_+ E_- v_{\Lambda-2N} = (E_- E_+ + H) v_{\Lambda-2N} = (r_N + \Lambda - 2N)\, v_{\Lambda-2N}\,. \tag{2.14}$$

This leads to

$$N^2 + (1-\Lambda)N - \Lambda = 0\,, \tag{2.15}$$

with solutions $N = -1$ and $N = \Lambda$. The dimension of the representation is thus

$$\dim = N + 1 = \Lambda + 1\,, \tag{2.16}$$

The eigenvalues of the diagonal generator of $sl(2)$ are integers, when the basis of generators (2.4) is used.

and most important: the value Λ has to be integer. The main result is thus that, in this basis, the eigenvalues of H are integers.

2.1.3 Preview: extending $sl(2)$

To get an idea of the logic behind the classification of Lie algebras, we will now first look at a concrete example, namely the extension of $sl(2)$ by at least one additional generator that commutes with H. We will see here how one is forced into introducing more and more ladder operators, but in such a way that the various commutators between the ladder operators are severely restricted. In fact, there are only three non-trivial ways to do this. In the next section we will then turn to a generalisation of this argument that classifies all finite-dimensional algebras.

Let us thus start from $sl(2)$, and try to see how to systematically extend this algebra to something larger. We start by adding a single generator H^2 which commutes with H^1 (these mutually commuting operators form what is known as the *Cartan subalgebra* or CSA). If we make this new generator commute not just with H^1 but also with the ladder operators $E^1_\pm$, we end up with a rather trivial system. So we will want to add at least one other generator, which does not commute with H^2. However, given that H^2 is hermitean, we will always need two extra generators, and these generators together with H^2 then necesssarily form an $sl(2)$.

Extending by only a diagonal generator H^2 is trivial, but if we want to add more, we need at least a pair E^2_+ and E^2_-, because of the hermiticity of H^2.

If we do not make these generators of the two $sl(2)$'s talk to each other, we end up with just a direct sum of two independent algebras, which is again rather trivial. Therefore, the first non-trivial result comes in when we make the generators of the two algebras have non-zero commutators with one another. For commutators with the CSA,

$$[H^i, E^j_\pm] = \pm A^{ji} E^j_\pm\,. \tag{2.17}$$

We must have $A^{ii} = 2$ to get the diagonal relations, but A^{12} and A^{21} are still arbitrary numbers. We cannot set them to zero because that would decouple the algebras completely.

In order to figure out what these numbers can be, we have to look at the result for the commutator of the step operators,

$$[E^1_\pm, E^2_\pm]\,. \tag{2.18}$$

These operators are in fact again step operators, as you can show that

$$[H^1, [E^1_\pm, E^2_\pm]] = \lambda [E^1_\pm, E^2_\pm] \tag{2.19}$$

for some value of λ. The essential ingredient here is the Jacobi identity (check this!). Another result that follows from repeated use of the Jacobi identity is that of the four commutators

$$[E^1_\pm, E^2_\pm] \quad \text{and} \quad [E^1_\mp, E^2_\pm]\,, \tag{2.20}$$

we have to have at least two non-zero ones.

From the representation theory of $sl(2)$ we know that A^{ji} are integers. If we keep only a minimal choice,

$$[E^1_\pm, E^2_\pm] \neq 0\,, \quad [E^1_\mp, E^2_\pm] = 0\,, \tag{2.21}$$

and all higher order commutators zero, one can show (Jacobi identities again) that this corresponds to setting

$$A_2: \quad A^{12} = -1\,, \quad A^{21} = -1\,. \tag{2.22}$$

As indicated, this algebra is called 'A_2', but is also known as $sl(3)$. There turn out to be two other options, given by

$$\begin{aligned} B_2: \quad & A^{12} = -2\,, \quad A^{21} = -1\,, \\ G_2: \quad & A^{12} = -3\,, \quad A^{21} = -1\,. \end{aligned} \tag{2.23}$$

It is at this stage not clear that these are in fact the *only* choices, but we will see this in the following sections.

▸ ***See also:*** This down-to-earth approach can also be found in chapter 3 of [**?**].

2.1.4 Real subalgebras

We have so far discussed complex algebras. That is to say, we were allowed to take *complex* linear combinations of the generators of the algebra. For the example of $sl(2)$, the three generators H, E_+ and E_1 given in matrix representation in (2.3) can thus be multiplied with any complex number, to produce any arbitrary matrix with complex elements, subject only to the condition that the trace vanishes. This algebra is more appropriately called $sl(2,\mathbb{C})$. For this algebra you could have started with some arbitrary other basis obtained as a complex linear combination of the basis generators which we chose; this clearly does not make any difference.

However, it is of course possible to look at restrictions of this algebra which are obtained by restricting the coefficients to real values. Once you do that, you have to be careful about which generators you take. If you stick to the generators in (2.3), an arbitrary linear combination with real coefficients give you all real traceless matrices. This algebra is called $sl(2,\mathbb{R})$.

The algebras $sl(2,\mathbb{R})$ and $su(2)$ are the two real subalgebras (or 'real forms') of $sl(2,\mathbb{C})$. As real algebras, they are inequivalent.

You can also choose to first go to a basis formed by the matrices (1.14), and then take real combinations of those to form an arbitrary element in the real algebra. This gives what is known as $su(2)$. This is a different algebra, with structure constants which differ from $sl(2,\mathbb{R})$ only because we restrict ourselves to real linear combinations.

2.1.5 Simple, semi-simple, ideals

2.2. Cartan classification and Dynkin diagrams

The general structure of a (simple) Lie algebra is given by a number r of mutually commuting generators H^i, together with other operators which are eigenvectors of H^i in the adjoint representation. The number r is called the *rank* of the algebra. We will start by slightly relaxing normalisation conditions on the generators, so that we have, apart from the trivial brackets between the Cartan subalgebra generators H^i, also the basic defining relations

The rank of an algebra is the number of CSA generators.

Eigenvalues of the CSA generators are called *weights*, or *roots* when they concern the adjoint representation.

$$[H_i, E_\alpha] = \alpha_i\, E_\alpha\,. \tag{2.24}$$

These α_i are called the *weights* of the adjoint representation, and also called *roots*. Because the E_α are not hermitian,

$$[H_i, E_\alpha^\dagger] = -\alpha_i E_\alpha^\dagger , \tag{2.25}$$

we see that if α is a root, so is $-\alpha$. The corresponding two operators are raising and lowering operators. A key ingredient which we have seen in the example is that the commutator of two ladder operators is itself a ladder operator,

Roots always come in pairs, with opposite sign.

$$[H_i, [E_\alpha, E_\beta]] = (\alpha + \beta)^i [E_\alpha, E_\beta] . \tag{2.26}$$

This follows from a Jacobi identity. Making this consistent with (2.24) requires that such a commutator is a multiple of the generator $E_{\alpha+\beta}$,

$$[E_\alpha, E_\beta] = c_{\alpha,\beta} E_{\alpha+\beta} . \tag{2.27}$$

Similarly, when $\alpha = -\beta$, we see from (2.26) that the commutator has to be a linear multiple of the CSA generators,

$$[E_\alpha, E_{-\alpha}] = d_i H^i . \tag{2.28}$$

Finally, when $(\alpha + \beta)_i$ is not an eigenvalue of H_i (in more technical terms: when $\alpha + \beta$ is not a root), then the commutator vanishes,

$$[E_\alpha, E_\beta] = 0 \quad \text{if } \alpha + \beta \notin \Phi. \tag{2.29}$$

A basis of this type is called a *Cartan-Weyl basis*.

We have also seen that not all Cartan-Weyl bases are equally nice. In particular, the Cartan-Killing form

$$(X, Y) := \mathrm{Tr}(\mathrm{ad}\, X\ \mathrm{ad}\, Y) \tag{2.30}$$

is not always diagonal on the Cartan subalgebra. However, you can always choose your generators in this subalgebra so that this 'metric' becomes

$$\mathrm{Tr}(H_i H_j) = \lambda \delta_{ij} . \tag{2.31}$$

Moreover, we can always normalise the ladder operators in the same way,

$$\mathrm{Tr}(E_{-\beta} E_\alpha) = \delta_{\alpha\beta} \tag{2.32}$$

We will use this choice as it makes many formulas much simpler and many intermediate steps redundant. With this choice, you can show that the d_i in (2.33) are actually equal to the components of the root, i.e.

$$[E_\alpha, E_{-\alpha}] = \alpha_i H^i . \tag{2.33}$$

There are many other bases used in the literature, but dealing with a non-trivial Killing form is rather a lot of extra work.

For every root α, there is an $sl(2)$ algebra (we have seen this explicitly for the three roots of $sl(3)$). In order to get this subalgebra in the canonical form you have to rescale the generators by factors involving the root and the norms of the root,

To every root (*not* only the simple ones!) is associated an $sl(2)$.

$$\tilde{E}_\pm = |\alpha|^{-1} E_{\pm\alpha} , \quad \tilde{H} = 2|\alpha|^{-2} \alpha \cdot H . \tag{2.34}$$

The basis obtained in this way is called a *Chevalley basis*. Because of the norms, it is here obviously quite useful that we are in a basis where the Killing form is diagonal.

We now need to bring a little bit more structure into the system of roots before we can continue. In general, given a set of roots Φ, it is always possible to choose a

subset Φ^+ of them, such that two conditions hold: if $\alpha \in \Phi^+$ then $-\alpha \notin \Phi^+$, and if $\alpha, \beta \in \Phi^+$, then $\alpha + \beta \in \Phi^+$. Essentially, this just amounts to picking a basis in root space (and there are often many ways to do this). The roots we choose in this way are called *positive* roots. Those elements $\alpha \in \Phi^+$ which cannot be written as the sum of positive roots are called *simple* roots. These simple roots satisfy the important property that the difference of two simple roots is not a root (show this!).

The difference of two simple roots is *not* a root.

The key idea will now be the following. We will start from some generator corresponding to a particular root β. We can then ladder with the $\tilde{E}_\pm$ (as constructed above) for a particular root, just like we did when we discussed $sl(2)$ representation theory. We found there that the eigenvalues of the $\tilde{H}$ operator have to be integers. Translated to our problem here, and taking care of the normalisation (2.34) that was required to obtain a canonically normalised $sl(2)$ algebra, this means that

$$2\frac{\langle \alpha, \beta \rangle}{\langle \alpha, \alpha \rangle} \in \mathbb{Z}\,. \tag{2.35}$$

However, since β is a root, you can also apply this with α and β exchanged. We can get a better understanding for what this means by observing that the inner product between two vectors is expressible in terms of the cosine of the angle between them,

$$\cos^2\theta_{\alpha\beta} = \frac{\langle \alpha, \beta \rangle^2}{\langle \alpha, \alpha \rangle \langle \beta, \beta \rangle} \in \frac{mm'}{4}\,. \tag{2.36}$$

The cosine square cannot be large than one. Moreover, we have to exclude the case where the cosine equals one, because that implies that the two roots are equal or opposite, which cannot happen because it would mean that one root occurs multiple times. Thus, there are only four possibilities,

mm′	θ
0	$\frac{\pi}{2}$ $(90°)$
1	$\frac{2\pi}{3}$ $(120°)$
2	$\frac{3\pi}{4}$ $(135°)$
3	$\frac{5\pi}{6}$ $(150°)$

(2.37)

A string of roots can thus never contain more than four roots.

The key quantities to classify a Lie algebra are thus these various integers (2.35). We have actually already given these a name before: when we extended $sl(2)$ to $sl(3)$, we implicitly used generators in a nice $sl(2)$ normalisation, and then called the eigenvalues of the H's the Cartan matrix elements. We thus now have a more formal definition of the Cartan matrix in terms of simple roots,[1]

$$A^{ij} := \frac{2\langle \alpha_i, \alpha_j \rangle}{\langle \alpha_j, \alpha_j \rangle}\,. \tag{2.38}$$

These elements are actually constrained further. First of all, the inner product between two different simple roots is always negative (see also below). Therefore, the non-diagonal elements of the Cartan matrix are negative. In addition, the Schwarz inequality tells us that

$$\langle \alpha_i, \alpha_j \rangle \leq \langle \alpha_i, \alpha_i \rangle \langle \alpha_j, \alpha_j \rangle \tag{2.39}$$

so that (again, the maximal case is trivial)

$$A_{ij}A_{ji} < 4\,. \tag{2.40}$$

[1] Pay attention to the order of the indices; the Cartan matrix is in general not symmetric.

The only choices for off-diagonal elements and their transpose partners are thus

$$\begin{aligned} A_{12} &= -1 \quad \text{and} \quad A_{21} = -1\,, \\ A_{12} &= -2 \quad \text{and} \quad A_{21} = -1\,, \\ A_{12} &= -3 \quad \text{and} \quad A_{21} = -1\,. \end{aligned} \tag{2.41}$$

Finally, since the matrix is built from α_i which should be linearly independent, we know that $\det A \neq 0$.

The determinant condition is a bit awkward to implement, and in fact eliminates many possibilities which are still left open by the conditions above (some examples will be seen in the tutorial problems). Working out which combinations are allowed is not particularly enlightening, so we will just jump straight to the result. This is most easily expressed in terms of graphical rules, which lead to the so-called *Dynkin diagrams*.

[KP: diagrams and rules to be added; see any decent book on Lie algebras]

2.2.1 Root systems

The logic that led us to conclude that the Cartan matrix elements are integers can in fact be played with an arbitrary root,μ, not just the simple ones. A key ingredient is that the lowest and highest weight of an $sl(2)$ representation are $-\Lambda$ and Λ respectively. For a generic weight μ in a string of roots, this leads to[2]

$$2\frac{\langle \mu, \beta \rangle}{\langle \alpha, \alpha \rangle} = m - p\,, \tag{2.42}$$

where m is the number of steps you can act still with a E_- lowering operator and p the number of steps that you can make with an E_+ raising operator before the root string ends. This is not particularly useful (for instance, a 'zero' just tells you that you are in the middle of the string, not how long it is on either end). However, if you know that m or p is zero, then this expression determines the length of the root chain. This is true for instance when you start with μ being a simple root, as the difference of two simple roots is not a root, and so $m = 0$ in that case. As a side result, this also immediately shows that inner products between simple roots are always negative.

The only multiple of a root α which is also a root is $-\alpha$.

[KP: laddering procedure to be added; discussed in detail on the board]

Once all roots (the simple as well as the non-simple ones) have been obtained, we can use (2.34) to construct all generators which explicitly generate all the $sl(2)$ subalgebras for us.

This laddering procedure which we have just described in terms of roots can also be described directly at the level of the algebra, and is known as the *Chevalley-Serre relations*. They read

$$\begin{gathered} [H^i, H^j] = 0\,, \quad [H^i, E^i_\pm] = \pm A^{ji} E^j_\pm\,, \quad [E^i_+, E^j_-] = \delta_{ij} H^i\,, \\ (\mathrm{ad}\, E^i_\pm)^{1-A^{ji}} E^j_\pm = 0\,. \end{gathered} \tag{2.43}$$

Here the first line shows us the commutation relations between the generators associated to the simple roots, while the second line determines which commutators of step operators are kept non-zero. This mimicks what we have seen in the $sl(3)$ example earlier.

[2] This formula in fact has even wider range of applicability, because we can use it not just for the eigenvalues in the adjoint representation (the roots), but als also for more general representations. We will return to this issue in the next chapter.

2.3. Rank-two examples

In order to see how the Cartan classification does its job, let us look in some detail at the three rank-two algebras that we introduced earlier: A_2, B_2 and G_2. Their Cartan matrices are given by

$$A_2: \begin{pmatrix} 2 & -1 \\ 1 & 2 \end{pmatrix}, \quad B_2: \begin{pmatrix} 2 & -2 \\ -1 & 2 \end{pmatrix}, \quad G_2: \begin{pmatrix} 2 & -3 \\ -1 & 2 \end{pmatrix}. \tag{2.44}$$

The first thing to do is to construct all the step operators, or equivalently, all the roots. For this we use the logic that led to (2.42). We start from a simple root, and determine whether we can add one of the other simple roots by computing (2.42). A general root which we encounter in this way is a linear combination of simple roots,

$$\beta = \sum_i k_i \alpha^i \,. \tag{2.45}$$

The sum of all k_i's is called the *level* of a root. Looking at (2.42), we see that what we need to do is to keep track of the numbers

$$\sum_j k_j A^{ji} \tag{2.46}$$

at every stage.

For B_2 we find ...

For G_2 the story is similar, but now the root chains are longer because of the presence of the '-3' in the Cartan matrix. We get

LiE:

```
> pos_roots(G2)
   [[1,0], [0,1] ,[1,1]
   ,[2,1], [3,1] ,[3,2]
   ]
```

$$\begin{array}{ccccc} & & \begin{pmatrix} 2 & -3 \\ -1 & 2 \end{pmatrix} & & \\ [2,-3] & & & [-1,2] & \\ & [1,-1] & & \alpha_1+\alpha_2 & m_1=m_2=1 \to p_2=2 \\ & [0,1] & & \alpha_1+2\alpha_2 & \\ & [-1,3] & & \alpha_1+3\alpha_2 & \\ & [1,0] & & 2\alpha_1+3\alpha_2 & \end{array} \tag{2.47}$$

In the third line, going from $\alpha_1+2\alpha_2$, you cannot go to $2\alpha_1+2\alpha_2$ because m for this step is zero: there is no root $2\alpha_2$, so $\alpha_1+2\alpha_2$ is actually at the bottom of an α_1 string, not at the first step.

3
Representation theory

The machinery of generalised ladder operators and their eigenvalues, the roots, has a much wider applicability than just the construction of algebras. It also allows us to discuss generic representations of those algebras. The logic that allowed us to determine the lengths of chains of roots does not only apply when starting with a root, but, as we have already briefly mentioned, also when we start with a more general set of eigenvalues of the Cartan subalgebra generators. In the present chapter we will see how this works in detail and look at a few physics applications.

3.1. Weight space

3.1.1 Weight vectors versus roots

Consider an arbitrary representation of an algebra. If we diagonalise the Cartan subalgebra generators H^i, we can characterise the representation by looking at the eigenvalues of these operators when acting on the vectors that span the representation space; compactly[1]

$$H|\mu\rangle = \mu|\mu\rangle \,. \tag{3.1}$$

The formula for the length of a root chain through a weight is exactly the same as for the special case when the weight is also a root,

$$2\frac{\langle \mu, \beta\rangle}{\langle \alpha, \alpha\rangle} = m - p \,, \tag{3.2}$$

This formula becomes useful when we know that adding a root to μ leads us to a weight which does not exist in the spectrum. This is exactly the case when μ is a highest weight, and in this case $p = 0$.

3.1.2 Building weight space

Following the logic of the previous subsection, we can construct the entire weight space given the highest weight, by laddering 'downwards' with the roots.[2] This is

[1] This expression can again be a bit confusing. We have r generators H, so H in this formula is an r-dimensional vector of operators (matrices) and μ is an r-dimensional vector of eigenvalues.

[2] If you would consider lowest-weight representations, then you could ladder upwards from the starting point and the procedure would mimick the construction of the space of roots more closely; the reason for looking at highest-weight representations is purely historical.

best made explicit with another example. Consider the representation with highest weight $[1,1]$. Its weight space follows as

$$\begin{array}{ccccc} & \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} & & & \\ & [1,1] & & \alpha_1+\alpha_2 & \\ [-1,2] & & [2,-1] & \alpha_2,\ \alpha_2 & \\ & [0,0] & & 0 & \\ [-2,1] & & [1,-2] & -\alpha_1,\ -\alpha_2 & \\ & [-1,-1] & & -\alpha_1-\alpha_2 & \end{array} \tag{3.3}$$

LiE:

To find the Dynkin labels of the adjoint representation,

```
> adjoint(A2)
   1X[1,1]
```

This gives all weights in the representation with highest weight $\alpha_1+\alpha_2$.

What the diagram above does not tell you is the multiplicity of each weight. You might have thought that since you can reach the $[0,0]$ weight in the middle of the diagram in two different ways, this leads to a multiplicity two. That is indeed the correct answer, but the wrong logic: using the same logic you would conclude that the lowest-weight state has multiplicity four, which is incorrect.

What you have to do in order to figure out the multiplicity is to look explicitly at the states corresponding to these weights, and determine whether the different ways of obtaining a state with the given weight lead to linearly independent states. As an example, it is not hard to show that the two states of weight $[0,0]$,

$$E_{-\alpha_1}E_{-\alpha_2}|\alpha_1+\alpha_2\rangle \quad \text{and} \quad E_{-\alpha_2}E_{-\alpha_1}|\alpha_1+\alpha_2\rangle \tag{3.4}$$

are linearly independent (this is because the two lowering operators do not commute). However, it is less clear a priori that the four different ways of obtaining states with weight $[-1,-1]$ are all linearly dependent. This requires repeated swapping around of lowering operators using commutation relations. Unfortunately this process is often laborious. There is a clever technique due to ... and a similarly clever formula do to Freudenthal; we may discuss these later if time permits.

▸ ***See also:*** Georgi is one of the few books discussing the multiplicity of weights in some detail; see around page 135. We will discuss the multiplicity issue again when we discuss Casimir operators later.

3.2. Tensor products

LiE:

```
> tensor(1X[1,1],1X[1,1],A2)
   1X[0,0] +1X[0,3] +2X[1,1]
   +1X[2,2] +1X[3,0]
```

3.3. Example: the eightfold way

▸ ***See also:*** The ideas behind the eightfold way are described in some detail in [**?**] from page 143 onwards, or in [**?**] section 3.6.

4
Assorted topics

4.1. Young tableaux for $sl(n)$

Young tableaux are graphical notations for highest-weight representations of $sl(n)$ (there exist modified tableau notations for other groups as well, see [**?**] for details). It is based on an idea we have seen earlier, namely that you can build representations using tensor products of *only* fundamental representations, suitably symmetrised or anti-symmetrised. This is a special property which only holds true for some algebras (and hence the Young tableau method is not as often seen for non-$sl(n)$ cases).

The $sl(3)$ example we have discussed reads, in Young tableau notation

$$\begin{array}{ccccccc} \square & \otimes & \square & = & \square\square & \oplus & \begin{array}{|c|}\hline \\ \hline \\ \hline \end{array} \\ \phi_i & & \phi_j & = & \phi_{(i}\phi_{j)} & + & \phi_{[i}\phi_{j]}\,. \end{array} \tag{4.1}$$

As indicated, each box on the left-hand side corresponds to a fundamental representation of dimension 3. The horizontal row of two boxes is the symmetrised tensor product, while the vertical row of boxes is the anti-symmetrised tensor product. Young tableaux generalise this idea to more complicated tensor products, in which the irreducible representations on the right hand side do *not* in general correspond to fully symmetrised or fully anti-symmetrised products.

To be more precise, a Young tableau with n boxes can be viewed as the tensor product of n fundamental representations, with each box being related to a fundamental index, with symmetrisation applied to every row and then anti-symmetrisation applied to each row (the result is then *no longer* symmetric in the indices in each row). An example of this procedure:

$$\begin{array}{|c|c|}\hline i & j \\ \hline k \\ \cline{1-1} \end{array} : \phi_i\phi_j\phi_k \rightarrow \frac{1}{2}\phi_i\phi_j\phi_k + \frac{1}{2}\phi_j\phi_i\phi_k$$
$$\rightarrow \frac{1}{3}\,\phi_i\phi_j\phi_k + \frac{1}{3}\,\phi_j\phi_i\phi_k - \frac{1}{3}\,\phi_k\phi_j\phi_i - \frac{1}{3}\,\phi_j\phi_k\phi_i\,. \tag{4.2}$$

The normalisation factor is chosen such that applying the combined symmetrisation and anti-symmetrisation operation twice gives the same result (in technical lingo: the Young projector is idempotent). Obviously, since you cannot anti-symmetrise more than n indices if they only take on n different values, the number of rows in a Young tableau for $sl(n)$ is at most n.

There is a rather simple way to compute the dimension of a representation from its associated Young tableau. This is easiest to explain at the level of a concrete

Tableau dimension formula.

example. Consider in $A_n = sl(n+1)$ the representation

. (4.3)

Its dimension is obtained using the ratio of two expressions. For $n = 3$ we have

$$d_Y = \frac{\begin{array}{|c|c|c|c|}\hline 4&5&6&7\\\hline 3&4\\\cline{1-2} 2\\\cline{1-1}\end{array}}{\begin{array}{|c|c|c|c|}\hline 6&4&2&1\\\hline 3&1\\\cline{1-2} 1\\\cline{1-1}\end{array}} = \frac{7!\times 4}{6!/5} = 140\,. \tag{4.4}$$

LiE:

```
> dim([2,1,1],A3)
   140
```

In the numerator, we start with $n+1$ and then fill boxes by following the rule that going to the right increases the number by one, and going down decreases by one. In the denominator, we fill each box with the sum of the number of boxes to the right and the number of boxes below it, plus one (this is known as the *hook formula*). To evaluate the diagrams, simply multiply the numbers in all the boxes. The result gives the number of independent components that you need in order to describe a tensor with the symmetries of the corresponding Young tableau.

A Young tableaux with b_m columns with m boxes can be labelled as $(b_1\, b_2\, b_3\, \ldots)$. In this notation, there is a direct relation between tableaux and the Dynkin labels of the corresponding highest weight representation. Here are some examples for $sl(3)$:

$[1,0]$		3
$[0,1]$		$\bar{3}$
$[1,1]$		8
$[2,0]$		6
$[0,2]$		$\bar{6}$
$[2,1]$		15

(4.5)

The last column indicates the dimension as computed using the dimension formula.

Another useful aspect of Young tableaux is that they can be used to decompose a tensor product of representations into irreducible ones. To do this, draw the two diagrams next to one another, and place in each box of the second diagram a symbol a in the first row, a b in the second row and so on. Now add the boxes of the second diagram to the first diagram, one by one, using the following rules at every step,

1. Each diagram thus constructed must be a Young tableau.
2. For A_n (i.e. $sl(n+1)$), no diagram can have more than $n+1$ rows (you would anti-symmetrise in more than $n+1$ indices which take on only $n+1$ different values, hence you would get zero).
3. Boxes with the same label must not appear in the same column.
4. At any given box position, define n_a to be the number of a's above and to the right of it (similarly for n_b and so on). Then we must have $n_a \geq n_b \geq n_c$ and so on.
5. Two tableaux of the same shape are counted as different only when the labelling is different.

You can understand some of these rules simply in terms of symmetries: the third rule, for instance, says that boxes which are symmetrised in the original diagram

cannot appear in anti-symmetrised locations in the resulting diagram. These rules go under the name of the *Littlewood-Richardson* algorithm. Let us see this in action using examples. The tensor product of two fundamentals of A_2 decomposes as

$$3 \otimes 3 = \square \otimes \boxed{a} = \square\boxed{a} \oplus \begin{array}{|c|}\hline \\ \hline a \\ \hline \end{array} = 6 \oplus 3\,. \tag{4.6}$$

The product of a fundamental times a conjugate fundamental gives instead

$$\bar{3} \otimes 3 = \begin{array}{|c|}\hline \\ \hline \\ \hline \end{array} \otimes \boxed{a} = \square\boxed{a} \oplus \begin{array}{|c|}\hline \\ \hline \\ \hline a \\ \hline \end{array} = 8 \oplus 1\,. \tag{4.7}$$

You could also compute this by exchanging the two factors. At the level of the Young tableau manipulations this leads to different intermediate steps, but the same end result:

$$\begin{aligned} 3 \otimes \bar{3} &= \square \otimes \begin{array}{|c|}\hline a \\ \hline b \\ \hline \end{array} \\ &\to \square\boxed{a} \oplus \begin{array}{|c|}\hline a \\ \hline \end{array} \\ &\to \begin{array}{|c|c|}\hline \\ \hline b \\ \hline \end{array}\!\!\boxed{a} \oplus \cancel{\square\boxed{a}\boxed{b}} \oplus \begin{array}{|c|}\hline a \\ \hline b \\ \hline \end{array} \oplus \cancel{\begin{array}{|c|c|}\hline \\ \hline a \\ \hline \end{array}\!\!\boxed{b}} = 8 \oplus 1\,. \end{aligned} \tag{4.8}$$

Cadabra:

```
\tab{#}::FilledTableau(dimension=3).
\tab{1}\tab{a}{b};
@lr_tensor!(%);
```

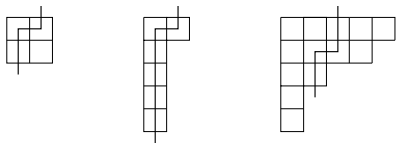
$$\begin{array}{|c|c|}\hline 1 & a \\ \hline b \\ \cline{1-1} \end{array} + \begin{array}{|c|}\hline 1 \\ \hline a \\ \hline b \\ \hline \end{array}\,;$$

The second diagram violates rule 4, while the fourth diagram violates rule 3.

▸ ***See also:*** Young tableaux are discussed in e.g. [**?**] section 8.4.

4.2. Bianchi identities

The more complicated symmetries are *multi-term* symmetries, such as the Ricci cyclic identity or the Bianchi identity, which relate a sum of terms with different index distributions. These symmetries are all manifestations of the so-called *Garnir* symmetries of Young tableaux [**?**]. These state that the sum over all anti-symmetrisation of boxes in a Garnir hook is identically zero. Examples of such Garnir hooks are given below,



which represent the Ricci cyclic identity, the Bianchi identity on a five-form and a more general Garnir symmetry, respectively. Applying a Garnir symmetry on a tensor produces a sum of tensors, which means that one can no longer restrict to the canonicalisation of tensor monomials.

There are, however, a few subtleties. Firstly, it would be prohibitively expensive to write down all terms in a Young-projected tensor. Instead, it is much more efficient to reduce the Young-projected forms by making use of the mono-term symmetries, which are easy to deal with using the methods of [**? ? ?**]. One thus obtains,

for e.g. the Riemann tensor,

$$R_{abcd} \to \frac{1}{3}\left(2\,R_{abcd} - R_{adbc} + R_{acbd}\right), \tag{4.9}$$

instead of the $(2!)^4 = 16$ terms which are produced by the Young projector. The expression on the right-hand side manifestly satisfies the cyclic Ricci identity, even if one only knows about the mono-term symmetries of the Riemann tensor. Using the projector (4.9) it is easy to show e.g. that $2\,R_{abcd}R_{acbd} = R_{abcd}R_{abcd}$. The monomial on the left-hand side maps to

$$R_{abcd}R_{acbd} \to \tfrac{1}{3}\left(R_{abcd}R_{acbd} + R_{abcd}R_{abcd}\right), \tag{4.10}$$

and similarly $R_{abcd}R_{abcd}$ maps to twice this expression, thereby proving the identity in a way which easily extends to much more complicated cases.

4.3. Branching rules and grand unified theories

One thing which the Dynkin classification of Lie algebras definitely makes more transparent is the issue of *subalgebras*. These are subsets of the generators of an algebra which form a closed set under the commutation relations. In physics, such subalgebras play an important role in the discussion of grand unified theories or GUTs, which are attempts to describe the strong, weak and electromagnetic force in one framework. This is a very rich topic of which we will here only discuss a few aspects to give you a flavour and a starting point to extend to more complicated situations.

As you will see in other modules in this MSc., the strong and weak nuclear force are related to $su(3)$ and $su(2)$ gauge symmetries respectively, while the electromagnetic force is described by a $u(1)$ gauge theory. It is a bit ugly to have these as three different groups, and many physicists have attempted to instead obtain these forces from one gauge theory which uses a larger gauge symmetry group.

In general, it may be possible to find the same subalgebras inside an algebra in multiple different ways. Consider for instance the $sl(2)$ subalgebras inside $sl(3)$. For $sl(3)$ we can write a matrix representation analogous to (2.3),

$$\begin{aligned}
H^1 &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad H^2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \\
E^1_+ &= \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad E^1_- = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \\
E^2_+ &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \quad E^2_- = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \\
E^\theta_+ &= \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad E^\theta_- = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.
\end{aligned} \tag{4.11}$$

One way to isolate an $sl(2)$ subalgebra in here is to simply use only H^1 together with the $E^1_\pm$ generators. This is what is called a *regular* embedding, as it simply uses a subset of the CSA generators togeter with the generators corresponding to a subset of the roots. An alternative embedding, however, is to use the 3-dimensional adjoint

representation of $sl(2)$ which we wrote down in (1.16). Those $T^{(a)}$ generators are expressible as linear combinations of the six raising and lowering operators written above, but this embedding cannot be viewed as restricting the CSA and the root system. Such embeddings are called *non-regular*. From the way we just described the difference, it should be obvious that regular embeddings are much easier to describe in the language of roots and weights. We will focus on these for the remainder of this section.

In the standard model, we have various fermion multiplets which transform under different representations of the $su(3) \times su(2) \times u(1)$ symmetry. We will not discuss that in detail, but here just take for granted that before electroweak symmetry breaking, these representations are

$$\begin{aligned}
\text{left-handed quark doublets } (u,d),(c,s),(t,b): &\ (3,2,\tfrac{1}{6})_L\,,\\
\text{right-handed up quarks } u,c,t: &\ (3,1,\tfrac{2}{3})_R\,,\\
\text{right-handed down quarks } d,s,b: &\ (3,1,-\tfrac{1}{3})_R\,,\\
\text{left-handed lepton doublet } (e^-,\nu_e): &\ (1,2,-\tfrac{1}{2})_L\,,\\
\text{right-handed electrons } e^-: &\ (1,1,-1)_R\,,\\
\text{right-handed neutrinos } \nu_e: &\ (1,1,0)_R\,.
\end{aligned} \tag{4.12}$$

There are of course three families of each of these; we have only written down one of them.

We can obviously embed $su(3) \times su(2)$ in $su(5)$, using a matrix embedding

$$U = \begin{pmatrix} U_3 & 0 \\ 0 & U_2 \end{pmatrix}. \tag{4.13}$$

Here $\operatorname{tr} U_3 = \operatorname{tr} U_2 = 0$. There is then a one-parameter family of traceless matrices left, of the form

$$Q = \begin{pmatrix} a & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 \\ 0 & 0 & a & 0 & 0 \\ 0 & 0 & 0 & b & 0 \\ 0 & 0 & 0 & 0 & b \end{pmatrix}, \tag{4.14}$$

where $3a + 2b = 0$. Such matrices generate the remaining $u(1)$. The map from the extended Dynkin diagram to the Dynkin diagram for the subgroups is

$$\textit{...graphic} \tag{4.15}$$

Restricting the 5,

$$\begin{array}{ccccc}
[& 1 & 0 & 0 & 0 \ ] \\
[& -1 & 1 & 0 & 0 \ ] \\
[& 0 & -1 & 1 & 0 \ ] \\
[& 0 & 0 & -1 & 1 \ ] \\
[& 0 & 0 & 0 & -1 \ ]
\end{array}
\quad \to \quad
\begin{array}{ccc}
[& 1 & 0 \ ] \\
[& -1 & 1 \ ] \\
[& 0 & -1 \ ] \\
[& 0 & 0 \ ] \\
[& 0 & 0 \ ]
\end{array}
\quad
\begin{array}{c}
[0] \\ [0] \\ [0] \\ [1] \\ [-1]
\end{array}
\,. \tag{4.16}$$

This means that the weight $z^n = [0\,0\,1\,0]^n$ is

With the choice $q = 1$ we get

$$\bar{5} \to (\bar{3},1,\tfrac{1}{3}) + (1,2,-\tfrac{1}{2})\,. \tag{4.17}$$

These correspond precisely to the (conjugate of the) down quarks and the left-handed leptons (electron and neutrino). An extended version of this analysis shows that in fact $\bar{5} + 10$ contains one entire family of the standard model. For more details see [**?**].