University of Bern

Institute for theoretical physics,
Albert Einstein center for theoretical physics

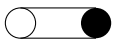
Bachelor Thesis

Graphical Tensor Product Reduction Scheme for the Lie Algebra so(5)

Supervisor:
Prof. Dr. U.-J. Wiese
Institute for theoretical physics
University of Bern

Author:
Bühlmann Patrick
13-101-126

July 2016



Abstract

If the dynamics of a physical system are invariant under certain transformations, the system has a symmetry. Symmetries play an important role in physics, since they allow us to simplify a problem and make qualitative statements and quantitative predictions. In this thesis continuous symmetries described by *Lie algebras* are investigated, in particular the so-called *so(5) Lie algebra*. When describing physical systems with Lie algebras it is vital to reduce tensor products of irreducible representations into sums of such representations. The graphical tensor product reduction scheme of J. P. Antoine and D. Speiser provides an algorithm to calculate these sums, without engaging in long and tedious calculations. The goal of this thesis is to work out this particular tensor product reduction scheme and then implement it in a Java-program.

Contents

1 Introduction

Sophus Lie ($*1842 - \dagger 1899$) was a Norwegian mathematician. He largely created the theory of continuous symmetry and applied it to the study of geometry. Lie's greatest achievement was the discovery that continuous transformation groups (now called Lie groups, after him) could be better understood by "linearizing" them and studying the corresponding tangential spaces which are spanned by so-called generators. The generators are subject to a linearized version of the group law and have the structure of what is today called a Lie algebra.
The Lie group $SO(n)$ which describes rotation in n-dimensional space, consists of all real, orthogonal $n \times n$ matrices O, with $\det O = 1$, and has a Lie algebra $so(n)$. The Lie group $SU(n)$ consists of all complex, unitary $n \times n$ matrices U with determinant 1. The well-known Lie algebra $su(2)$ describes, for example, quantum mechanical angular momentum and results from the Lie group $SU(2)$.
The Lie group $SO(5)$ has an underlying Lie algebra $so(5)$ which can be used to describe certain symmetries occurring in nature. This particular Lie algebra has been studied in attempts to unify the order parameters of antiferromagnetism and high-temperature superconductivity in condensed matter physics [1].
The Lie algebras $so(5) \simeq sp(2)$, $g(2)$, $su(3)$ and $so(4)$ are so-called rank 2 Lie algebras, where $so(4)$ is just the direct sum of two rank 1 $su(2)$ Lie algebras: $so(4) = su(2) \oplus su(2)$.
The Lie algebras $g(2)$ and $su(3)$ also play an important role in physics. $g(2)$ has been used in the context of string theory and supersymmetry [2]. The Lie algebra $su(3)$, on the other hand, has been used to describe flavour symmetries of up, down, and strange quarks ($su(3)_f$) as well as describing the color degree of freedom by which quarks couple to the gluon field ($su(3)_c$).
As one can see, rank 2 Lie algebras play an important role in physics. In all applications when working with Lie algebras it is important to reduce tensor products of irreducible representations into sums of such representations.
Consider, for example, the simple rank 1 Lie algebra $su(2)$. When working with a coupled system of a spin 2 and a spin $\frac{1}{2}$ particle, it is natural to "reduce" this product using the coupling rules:

$$\text{spin } 2 \otimes \text{spin} \frac{1}{2} = \text{spin} \frac{5}{2} \oplus \text{spin} \frac{3}{2} \tag{1}$$

Since every particle with spin S has $\{2S+1\}$ states, we can write this equation in terms of the dimension which denotes the number of states:

$$\{5\} \otimes \{2\} = \{6\} \oplus \{4\} \tag{2}$$

It is worth mentioning that the expression $\{2S+1\}$ for $su(2)$ describes a so-called $(2S+1)$ dimensional multiplet (or weight diagram) which can be visualized graphically. Since $su(2)$ is a rank 1 Lie algebra, the multiplet can be visualized one-dimensionally. For instance the multiplet of a spin 2 particle ($\{5\}$) can be presented as:

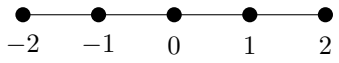


Figure 1: Multiplet of a spin 2 particle. The numbers denote the z-component of the spin.

Young tableaux provide a useful tool in order to reduce tensor products. However, in 1964 J. P. Antoine and D. Speiser described a graphical, much more economical method to reduce tensor products of rank 2 Lie algebras [3, 4]. Based on the fact that the multiplets of rank 2 Lie algebras are two-dimensional, they constructed a two-dimensional "landscape", which contains all the irreducible representations of a given rank 2 Lie algebra. Given two multiplets one wants to couple, the first one is placed into this landscape of irreducible representations, centered at the second multiplet. Taking into account the parity of the various sectors of the landscape and the multiplicities of the first multiplet one can easily calculate even higher-dimensional tensor products. In $so(5)$ one can couple for example a 10000-dimensional multiplet with a

10-dimensional one. Using the graphical method described by J. P. Antoine and D. Speiser the calculations are short and could even be done by hand (given a sufficiently large landscape):

$$\begin{aligned}\{10000\} \otimes \{10\} =& \{14080\} \oplus \{12320\} \oplus \{11340\} \oplus \{10240\} \oplus 2\{10000\} \\ & \oplus \{8960\} \oplus \{8360\} \oplus \{7980\} \oplus \{6720\}\end{aligned} \tag{3}$$

1.1 Structure and aim of this thesis

This thesis focuses mainly on Lie algebras, representations and the reduction of tensor products. In section 2 we provide a brief introduction to group theory, we give formal definitions and introduce the notations used throughout this thesis. We continue by applying the group theory to $so(5)$ and highlight the differences between this Lie algebra and other Lie algebras like $su(2)$ or $so(4)$. In section 4 we explain the idea behind coupling tensor products and describe the graphical tensor product reduction scheme. Section 5 is all about the graphical tensor product reduction scheme, which was described by J. P. Antoine and D. Speiser in 1964. The practical part of this thesis, namely the implementation of the previously mentioned algorithm, is described in section 6.

2 Group theory

Symmetries of a physical system are described by a mathematical structure known as a group. In the following a few essential facts about groups, Lie algebras, and representations will be given. Throughout this thesis we do not distinguish between contravariant and covariant vectors and follow the Einstein summation convention. *Lie groups* are denoted by capital letters, the corresponding *Lie algebras* are denoted by small letters, e.g., the Lie group SO(5) has the Lie algebra so(5). Furthermore we set $\hbar = 1$.

2.1 Groups

A **group** consists of a set G and an operation (denoted by $*$) on G with the following properties:

(G1) $* : G \times G \to G;\ (a,b) \mapsto *(a,b)\ \forall a,b \in G$

(G2) $(a * b) * c = a * (b * c)\ \forall a,b,c \in G$

(G3) $\exists e \in G$ such that $\forall a \in G : a * e = e * a = a$

(G4) $\forall a \in G\ \exists a^{-1} \in G$ such that $a * a^{-1} = a^{-1} * a = e$

If for all $a, a' \in G$ it holds that: $a * a' = a' * a$ then the group is called *Abelian*.
A group is called *continuous* if its group elements are functions of one or several continuous parameters, for example $G = \{a, b, c, d, \cdots\} = \{a(t), b(t), c(t), d(t)\}$[1].
If a group is continuous we need to impose some more properties on the group structure: Let $a \in G$ and t_1, t_2, t_3, t_4 continuous parameters then

$$\begin{aligned} a(t_1) * a(t_2) = a(t_3), &\qquad \Longrightarrow\ t_3 = f(t_1, t_2), \\ a(t_1)^{-1} = a(t_4), &\qquad \Longrightarrow\ t_4 = \hat{f}(t_1), \end{aligned}$$

where f and $\hat{f}$ are continuous functions. If the functions f and $\hat{f}$ are analytic the group is called a *Lie group*.
All these conditions assure that a Lie group has the geometrical structure of a differential manifold[2]. This leads to the fact that the properties of a Lie group can be understood by examining the immediate vicinity of one particular point on the manifold. Conventionally the chosen point is the identity element of the manifold. Elements of the corresponding *Lie algebra* can be obtained by determing the vectors lying in the tangent space at the identity. The examples which are the most common in physics are the infinite families of matrix groups: $GL(n), SL(n), O(n), SO(n), U(n), SU(n)$, and $Sp(n)$, together with the family of five exceptional Lie Groups $G(2), F(4), E(6), E(7)$, and $E(8)$. Before considering an important example, namely the Lie Group $SU(2)$, we want to give some formal definitions for the families of matrix groups:

$$\begin{aligned} Mat(n,\mathbb{F}) &= \{\text{set of } n \times n\text{-matrices with entries in } \mathbb{F}, \text{ where } \mathbb{F} \text{ is a field}\} \\ GL(n,\mathbb{C}) &= \{M \in Mat(n,\mathbb{C}) | det(M) \neq 0\} \\ SL(n,\mathbb{C}) &= \{M \in Mat(n,\mathbb{C}) | det(M) = 1\} \subset GL(n,\mathbb{C}) \\ O(n,\mathbb{R}) &= \{M \in Mat(n,\mathbb{R}) | MM^T = M^T M = \mathbb{1}\} \\ SO(n,\mathbb{R}) &= SL(n,\mathbb{C}) \cap O(n,\mathbb{R}) \\ U(n,\mathbb{C}) &= \{M \in Mat(n,\mathbb{C}) | MM^\dagger = M^\dagger M = \mathbb{1}\} \\ SU(n,\mathbb{C}) &= SL(n,\mathbb{C}) \cap U(n,\mathbb{C}) \\ Sp(2n,\mathbb{C}) &= \{M \in Mat(2n,\mathbb{C}) | M^T A M = A\} \\ \text{such that } A &= \begin{pmatrix} 0 & -\mathbb{1}_{n\times n} \\ \mathbb{1}_{n\times n} & 0 \end{pmatrix} \end{aligned}$$

[1] A group depending on r different parameters is called an *r-parameter group*.
[2] A *Differential Manifold* is a topological manifold with a global differential structure.

2.1.1 The Lie Group SU(2)

In order to make the transition from Lie groups to Lie algebras, we take a closer look at $SU(2)$ which is used to describe spatial rotations

$$SU(2) = \{U \in GL(2;\mathbb{C}) | UU^\dagger = U^\dagger U = \mathbb{1} | \det(U) = 1\}.$$

Let $U \in SU(2)$, then due to the restriction $UU^\dagger = U^\dagger U = \mathbb{1}$, we can write U as

$$U = \begin{pmatrix} u_0 + iu_3 & u_2 + iu_1 \\ -u_2 + iu_1 & u_0 - iu_3 \end{pmatrix}, u_i \in \mathbb{R}.$$

The second restriction yields

$$u_0^2 + u_1^2 + u_2^2 + u_3^2 = 1. \tag{4}$$

which means that there is a bijective correspondence between $SU(2)$ and S^3, the three-dimensional sphere embedded in $\mathbb{R}^4$. Therefore we conclude that the group manifold of $SU(2)$ is S^3. In order to determine some elements of the corresponding Lie algebra $su(2)$ we consider a matrix U near the identity[3]:

$$U = \mathbb{1} + i\epsilon A. \tag{5}$$

Using

$$\det(U) = 1 + i\epsilon \mathrm{tr}(A) + O(\epsilon^2) \stackrel{!}{=} 1,$$
$$UU^\dagger = \mathbb{1} + i\epsilon(A - A^\dagger) + O(\epsilon^2) \stackrel{!}{=} \mathbb{1}.$$

We conclude that the Lie algebra $su(2)$ can be described in terms of Hermitian, traceless 2×2-matrices. One can easily verify that the well-known *Pauli matrices* span $su(2)$ and are therefore known as the *generators* of the Lie algebra $su(2)$.

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{6}$$

2.2 Lie algebras

As already mentioned, the vectors lying in the tangent space at the identity element make up the Lie algebra of the group. Computations in the Lie algebra are often easier than those in the group and provide much of the same information.
A **Lie algebra** consists of a vector space g and an inner product (Lie bracket)

$$[.,.] : g \times g \to g, \tag{7}$$

which has the following properties for all $\lambda, \mu, \epsilon, \nu \in \mathbb{C}$ and $Z_i \in g$:

(L1) Bi-linearity:
$[\lambda Z_i + \mu Z_j, \epsilon Z_k + \nu Z_l] = \lambda\epsilon[Z_i, Z_k] + \lambda\nu[Z_i, Z_l] + \mu\epsilon[Z_j, Z_k] + \mu\nu[Z_j, Z_l]$

(L2) Anti-symmetry $[Z_i, Z_j] = -[Z_j, Z_i]$

(L3) Jacobi-Identity: $[Z_i, [Z_j, Z_k]] + [Z_k, [Z_i, Z_j]] + [Z_j, [Z_k, Z_i]] = 0$

A Lie-Algebra g is spanned by a basis which consists of a set of Hermitian generators X^α, i.e, $X^\alpha = (X^\alpha)^\dagger$. The index α runs from $1, \ldots, n_g$, where n_g denotes the dimension of the algebra. Any $Z \in g$ can be written as a real linear combination of generators: $Z = \omega^a X^a$, $\omega^\alpha \in \mathbb{R}$. A Lie algebra is specified by its structure constants

$$[X^a, X^b] = if_{abc}X^c, \qquad f_{abc} \in \mathbb{R}, \tag{8}$$

[3]The factor i is chosen conventionally in physics. As a result we get Hermitian generators.

such that for $Z_1 = \omega_1^a X^a$, $Z_2 = \omega_2^a X^a$ the Lie bracket reads

$$[Z_1, Z_2] = \omega_1^a \omega_2^b [X^a, X^b] = \omega_1^a \omega_2^b i f_{abc} X^c. \tag{9}$$

The Lie algebra $su(2)$ with its corresponding structure constants is given by

$$S_i = \frac{1}{2}\sigma_i, \qquad [S_i, S_j] = i\epsilon_{ijk} S_k. \tag{10}$$

A *subalgebra* is a vector space $\tilde{g} \subset g$ which is closed under the Lie bracket

$$[X^i, X^j] \in \tilde{g}, \qquad \forall X^i, X^j \in \tilde{g}. \tag{11}$$

An *ideal* $\tilde{g} \subset g$ is a subalgebra which is closed and "absorbing" which means

$$[X^i, X^j] \in \tilde{g}, \qquad \forall X^i \in \tilde{g}, \forall X^j \in g. \tag{12}$$

The *center* is an important ideal and consists of all generators in g that commute among themselves and with all other generators in g. It is strictly defined as

$$\mathbb{Z}[g] = \{Z \in g | [Z, g] = 0\}. \tag{13}$$

An *Abelian subalgebra* Λ is an algebra such that

$$[X^i, X^j] = 0, \qquad \forall X^i, X^j \in \Lambda \subset g. \tag{14}$$

If an algebra does not have any non-trivial ideals it is called *simple* and *semi-simple* if it has no Abelian ideals. Note that a semi-simple algebra may contain a non-Abelian ideal.
We call g a *direct sum*, if $g_1, g_2 \subset g$ are two commuting algebras satisfying:

$$\begin{aligned} [X^a, X^b] = i f_{abc} X^c \in g_1, & \qquad \forall X^a, X^b \in g_1, \\ [K^m, K^n] = i l_{mno} K^o \in g_2, & \qquad \forall K^m, K^n \in g_2, \\ [X, K] = 0, & \qquad \forall X \in g_1 \ , \forall K \in g_2. \end{aligned} \tag{15}$$

The commuting property is denoted by $g_1 \cap g_2 = \varnothing$, the direct sum by

$$g = g_1 \oplus g_2. \tag{16}$$

2.3 Cartan-Weyl basis

Given a Lie algebra with its basis $\{X^1, X^2, \cdots, X^{n_g}\}$ we can always transform it into the standard Cartan-Weyl form by a change of basis. This Cartan-Weyl basis is useful since it allows us to describe a Lie algebra g in terms of ladder and weight operators. Here we want to give a quick guide on how to construct the Cartan-Weyl basis. First we need to find the maximal set of commuting Hermitian generators $H^i, i \in \{1, \cdots, r\}$, where r is the *rank* of the algebra

$$[H^i, H^j] = 0, \qquad \forall i, j \in \{1, \cdots, r\}.$$

This set of generators is called the *Cartan subalgebra* h. The generators of this subalgebra can be diagonalized simultaneously and are the so-called *weight operators*. The remaining generators E^α are chosen as linear combinations of the X^i's such that they satisfy the following eigenvalue equation

$$[H^i, E^\alpha] = \alpha^i E^\alpha, \qquad \forall i \in \{1, \cdots, r\}. \tag{17}$$

The generator E^α is called a *ladder operator*, and $\alpha = (\alpha^1, \cdots \alpha^r)$ is known as a *root*. We can see that taking the Hermitian conjugate of equation (17) implies that $(-\alpha)$ is necessarily a root whenever α is a root

$$E^{-\alpha} = (E^\alpha)^\dagger. \tag{18}$$

Here we have already given a very general definition of the Cartan-Weyl basis. The concepts we are going to use throughout this thesis are the ones of the Cartan subalgebra and the already mentioned ladder and weight operators.

2.3.1 Cartan-Weyl basis for su(2)

Consider again the Lie algebra $su(2)$. Going into the Cartan-Weyl form means

$$\{S_1, S_2, S_3\} \to \{S_+, S_-, S_3\}, \tag{19}$$

with: $S_+ = S_1 + iS_2$, $S_- = S_1 - iS_2$, such that the new commutation relations read

$$[S_\pm, S_\mp] = \pm 2S_z, \qquad [S_3, S_\pm] = \pm S_\pm. \tag{20}$$

In this basis we can see that the ladder operators $S_\pm$ can be interpreted as movements between the states of a multiplet. Consider a state $|m_S\rangle$ in an arbitrary multiplet $\{2S+1\}$, such that $m_S \in \{-S+1, \cdots, S-1\}$. Then the ladder operators transform the eigenstates such that

$$S_+ |m_S\rangle \sim |m_S + 1\rangle, \qquad S_- |m_S\rangle \sim |m_S - 1\rangle. \tag{21}$$

The weight operator, on the other hand, gives the eigenvalue (or "weight") of a given state

$$S_3 |m_S\rangle = m_S |m_S\rangle. \tag{22}$$

Graphically one can represent the actions of the operators like in figure 2:

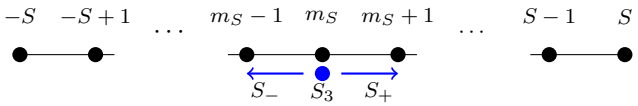


Figure 2: $su(2)$-multiplet with S_+, S_-, S_3.

This concept of representing ladder operators graphically generalizes even to the level of $so(5)$.

2.4 Representation

A **representation** of a Lie algebra means that we can associate to every $X^\alpha \in g$ a square matrix $\tilde{X}^\alpha$, such that the set of matrices obey the same algebraic relations as the fundamental generators, i.e.,

$$\omega^i X^i + \omega^j X^j \to \omega^i \tilde{X}^i + \omega^j \tilde{X}^j, \tag{23}$$

$$[X^i, X^j] = i f_{ijk} X^k \to [\tilde{X}^i, \tilde{X}^j] = i f_{ijk} \tilde{X}^k. \tag{24}$$

A representation is said to be *irreducible* if the matrices representing the elements of a given Lie algebra g cannot all be brought in a block diagonal form by a change of basis.
There are several ways of representing generators in terms of matrices due to the fact that the only given restriction is to satisfy the commutation relations. We have already seen an example of a representation:
The Pauli matrices are the generators of the *spinor representation* of $su(2)$. The spinor representation is the fundamental representation in terms of complex 2×2-matrices, for an object with quantum numbers $\{-\frac{1}{2}, \frac{1}{2}\}$. For the Lie algebra $su(2)$ the spinor representation can be obtained by writing a $U \in SU(2)$ infinitesimally close to the unit element (see equation (5)).
An n-dimensional *vector representation* arises naturally from the corresponding Lie Group $SO(n)$ by writing a group member infinitesimally close to the identity: $O = \mathbb{1} + i\epsilon A$. As a result the matrices are all traceless, Hermitian and purely imaginary. This representation is quite useful since it gives a direct relation between the Lie Algebra and its corresponding Lie Group. One can associate an arbitrary element $H = \omega^\alpha \tilde{X}^\alpha$ of the Lie algebra, written in the vector representation, with its corresponding counterpart in the Lie group by the relation

$$O = \exp(iH) = \exp(i\omega^\alpha \tilde{X}^\alpha). \tag{25}$$

The *adjoint representation* is an n_g-dimensional representation which encodes all the commutation relations between the generators. The adjoint representation is defined as

$$\mathrm{ad}(X^\alpha)X^\beta = [X^\alpha, X^\beta] \qquad \Leftrightarrow (\tilde{X}^\alpha)_{\beta\gamma} = -if_{\alpha\beta\gamma}. \tag{26}$$

Since working with the adjoint representation can be quite tedious for high-dimensional Lie algebras[4] it won't come into play in this thesis.
We complete our discussion by mentioning once again, that the fundamental structure of a given Lie algebra is encoded in its commutation relations. Representations provide a useful tool to describe the physics of the algebra or to unlock the mathematical behaviour of the corresponding group.

2.5 Comment on multiplets

Up to now we often used the notion "multiplet" without specifying what it is exactly. Here we want to catch up on this.
We start with a given symmetry group, its underlying algebra, and the set of operators in the Cartan-Weyl form which belong to the algebra. Consider now an invariant subspace of the whole Hilbert space consisting of all states $|\Psi_i\rangle$. Being an invariant subspace with respect to the given symmetry group means that all states reproduce themselves by acting with its ladder operators. One can also say that the operators transform all states of the invariant subspace among themselves.
A **multiplet** is an irreducible invariant subspace, i.e., a subspace which does not contain any other invariant subspace. In order to specify this and the idea of representations we give two examples: a spin one-half fermion and a spin one boson whose spin-properties are described by $su(2)$. A spin one-half fermion can have two possible states: $|m_s\rangle = |\frac{1}{2}\rangle$ or $|m_s\rangle = |-\frac{1}{2}\rangle$. A possible representation is the 2-dimensional spinor representation in the Cartan-Weyl form:

$$S_+ = S_1 + iS_2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \qquad S_- = S_1 - iS_2 = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \qquad S_3 = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{27}$$

This representation yields the following multiplet and vice versa:

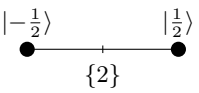


Figure 3: Two dimensional multiplet {2}.

Analogously, for a spin one boson one finds the following 3×3-representation to the 3-dimensional multiplet:

$$S_+ = \begin{pmatrix} 0 & \sqrt{2} & 0 \\ 0 & 0 & \sqrt{2} \\ 0 & 0 & 0 \end{pmatrix} \qquad S_- = \begin{pmatrix} 0 & 0 & 0 \\ \sqrt{2} & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{pmatrix} \qquad S_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{28}$$

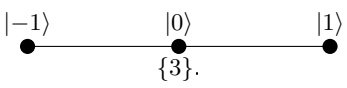


Figure 4: Three dimensional multiplet {3}.

We conclude that we can associate a D-dimensional multiplet $\{D\}$ with every set of $D \times D$-matrices obeying the commutation relations. This relation also holds true the other way around.

[4]for example the adjoint representation for so(5) is 10-dimensional.

Since there is a bijective correspondence between representations and multiplets we do not strictly distinguish between those two objects. Generally, when talking about a representation we mean a $D \times D$-matrix, talking about a multiplet we mean a graphical object as in figure 3 or figure 4.

2.6 Trivial and non-trivial duality

In the case of $su(2)$ we have two different types of multiplets: Those with trivial duality meaning that there is a state belonging to the eigenvalue $|m_s\rangle = |0\rangle$ and those with non-trivial duality meaning that there is no state $|0\rangle$. Trivial examples are the multiplets of a spin one-half particle $\{2\}$ and a spin one particle $\{3\}$ (see figure 3 and figure 4).
We will see later on that this concept also works for $so(5)$-multiplets. Examples of multiplets with non-trivial duality are given by the 4-dimensional spinor representation $\{4\}$ or $(1,1) = \{16\}$ (see figure 9). Some multiplets with trivial duality are given by the five dimensional vector representation $\{5\}$ or the adjoint representation $\{10\}$ (see figure 10).

2.7 The Lie Algebra so(4)

Since we will use $so(4) \subset so(5)$ we want to determine some interesting facts about the rank 2 Lie algebra $so(4)$. First of all, by calculating the generators in the vector representation, starting from the Lie Group SO(4) yields the following possible set of generators

$$A^1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -i & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \qquad A^2 = \begin{pmatrix} 0 & 0 & i & 0 \\ 0 & 0 & 0 & 0 \\ -i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \qquad A^3 = \begin{pmatrix} 0 & -i & 0 & 0 \\ i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$B^1 = \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ i & 0 & 0 & 0 \end{pmatrix}, \qquad B^2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & 0 & 0 \\ 0 & i & 0 & 0 \end{pmatrix}, \qquad B^3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{pmatrix},$$

following the commutation relations:

$$[A^i, A^j] = i\epsilon_{ijk}A^k, \qquad [A^i, B^j] = i\epsilon_{ijk}B^k, \qquad [B^i, B^j] = i\epsilon_{ijk}A^k.$$

By a change of basis such that

$$X_i = \frac{1}{2}(A^i + B^i), \qquad Y_i = \frac{1}{2}(A^i - B^i),$$

one obtains

$$[X_i, X_j] = i\epsilon_{ijk}X_k, \qquad [X_i, Y_j] = 0, \qquad [Y_i, Y_j] = i\epsilon_{ijk}Y_k. \tag{29}$$

In this new form one can see that $so(4)$ can be decomposed into a direct sum of two $su(2)$ subalgebras

$$so(4) = su(2)_x \oplus su(2)_y. \tag{30}$$

We conclude that the semi-simple Lie algebra $so(4)$ is the direct sum of the two simple Lie algebras $su(2)$. One can think of an $su(2)$ algebra "facing in the x-direction" and another one "facing in the y-direction", acting independently of each other. As a result of this decomposition, $so(4)$ has two 2-dimensional spinor representations, but also a 4-dimensional vector representation:

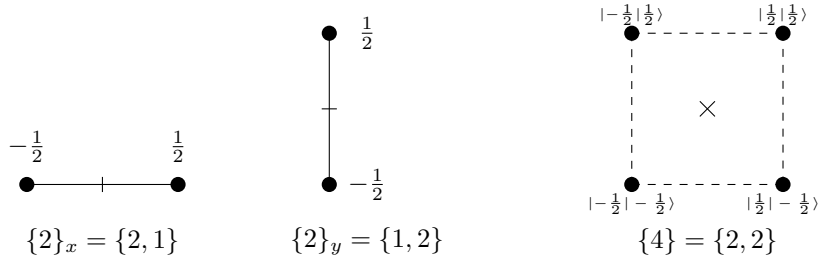


Figure 5: Spinor and vector representations of $so(4)$.

2.8 Universal covering group

Using Lie algebras we reach a deeper understanding about the local structure of a Lie group, namely by considering the tangential space near the unit element. The local structure can be the same for different groups, which have globally different manifolds. Therefore any Lie algebra has exactly one Lie group which provides information about the global structure, the so-called *covering group*. The covering groups for the Lie algebras $SO(n)$ are called *Spin-groups*: $Spin(n)$. For instance, one can easily show that the Lie algebras $so(3)$ and $su(2)$ are isomorphic[5] to each other, i.e., their corresponding Lie groups $SO(3)$ and $SU(2)$ look the same in an infinitesimal vicinity of the unit element. Although the Lie algebras $so(3)$ and $su(2)$ are isomorphic to each other, their corresponding Lie groups $SO(3)$ and $SU(2)$ are not. Considering the center (equation (13)) of both Lie groups one can easily see that no isomorphism can be built between them. The center of $SU(2)$ is $\mathbb{Z}(2) = \{\mathbb{1}, -\mathbb{1}\}$ and the center of $SO(3)$ is just $\{\mathbb{1}\}$, which shows profound differences in the group structure. Nevertheless $SU(2)$ and $SO(3)$ are closely related. The adjoint representation of $SO(3)$ is related to the spinor representation by

$$\begin{aligned} &f_{ab} : SU(2) \to SO(3), \\ &\qquad U \mapsto f(U)_{ab} = O_{ab} = \frac{1}{2} tr(U\sigma^a U^\dagger \sigma^b), \qquad \forall a, b \in \{1,2,3\}. \end{aligned} \tag{31}$$

The unit element of $SU(2)$ is mapped to

$$O_{ab} = \frac{1}{2} tr(\sigma^a \sigma^b) = \delta_{ab},$$

and the inverse $U^\dagger$ is mapped to

$$\frac{1}{2} tr(U^\dagger \sigma^a U \sigma^b) = \frac{1}{2} tr(U\sigma^b U^\dagger \sigma^a) = O_{ba} = O_{ab}^T,$$

where we used the cyclic property of the trace. The mapping f is not one-to-one, since both U and $-U$ are both mapped to the same group element $O \in SO(3)$. By straightforward calculations one can see that the kernel of the map f is $\mathbb{Z}(2)$, the center of $SU(2)$. Using isomorphism theorems for groups one obtains that

$$SO(3) \simeq SU(2)/\mathbb{Z}(2) \simeq S^3/\mathbb{Z}(2). \tag{32}$$

We conclude that the group manifold of $SO(3)$ is the sphere S^3 where all anti-podal points are identified with each other. Hence $SU(2)$ covers $SO(3)$ 2 times, i.e., the covering group of $SO(3)$ is $Spin(3) = SU(2)$.
The center of $SO(4)$ is $\{\mathbb{1}, -\mathbb{1}\}$ and its universal covering group can be shown to be: $Spin(4) = SU(2) \otimes SU(2)$, which has center $\mathbb{Z}(2) \otimes \mathbb{Z}(2)$. The Lie group $SO(5)$ has center $\{\mathbb{1}\}$ but its covering group $Spin(5) = Sp(2)$ has center $\mathbb{Z}(2)$.
Universal covering groups allow us to generalize the concept of *trivial* and *non-trivial duality*

[5] Isomorphisms are going to be denoted with $\simeq$, e.g., $so(3) \simeq su(2)$.

depending on the center of the universal covering group.
For instance in the case of $so(3) \simeq su(2)$ and $so(5) \simeq sp(2)$ the universal covering groups $Spin(3) = SU(2)$ and $Spin(5) = Sp(2)$ have center $\mathbb{Z}(2)$. As a result multiplets can be divided into those with trivial and those with non-trivial duality (see subsection 2.6). Due to this fact also the landscape, we will encounter latter on, can be built using two sublattices (see figure 14 and figure 16).
The center of $Spin(4) = SU(2) \otimes SU(2)$ is $\mathbb{Z}(2) \otimes \mathbb{Z}(2)$, consequently multiplets can be divided with respect to the $su(2)_x$ and the $su(2)_y$ duality. There are multiplets which have non-trivial duality with respect to both $su(2)$ subalgebras (for example, the vector representation of $so(4)$, see figure 5), multiplets which have trivial duality with respect to $su(2)_x$ but non-trivial duality with respect to $su(2)_y$ and so on, such that we finally obtain 4 different types of multiplets. As in the case of $su(2)$ and $so(5)$ this also has an effect on the landscape of $so(4)$: It is built by 4 sublattices, denoted by different colors in figure 15. Exact mathematical explanations referred to this subject can be found in [5].

3 Group theory of so(5)

The real-valued 5×5 orthogonal matrices O with determinant 1 obey $OO^T = O^T O = \mathbb{1}$ and form the Lie group SO(5) under matrix multiplication. Imposing these conditions one concludes that a general $O \in SO(5)$ has 10 free parameters, therefore $SO(5)$ is 10-dimensional[6] and so is its Lie algebra so(5). One can determine the generators in the vector representation using the linearization method by writing an element infinitesimally close to the unit element: $O = \mathbb{1} + i\epsilon A$. Similar to $so(4)$, the set of generators in the vector representation consists of all purely imaginary, Hermitian 5×5-matrices with trace 0. However, instead of using the vector representation, mathematics has shown us that there exists a 4-dimensional spinor representation for so(5). In the spinor representation we can build the matrices we need as a tensor product of Pauli matrices. This representation fully takes into account that $so(5)$ contains two $su(2)$ subalgebras: $su(2) \oplus su(2) = so(4) \subset so(5)$.

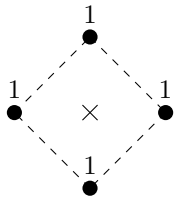


Figure 6: (1,0) = {4} in $so(5)$.

With the 4-dimensional spinor representation we can describe the 4-dimensional multiplet {4} given in figure 6, where the numbers denote the multiplicities, i.e., the number of states associated with each single point.

3.1 Spinor representation

Here we derive the 4-dimensional spinor representation for $so(5)$. Imposing the representation to be traceless and Hermitian, an arbitrary generator M in this representation space can be written as:

$$M = \begin{pmatrix} a_{11} & a_{12} + ib_{12} & a_{13} + ib_{13} & a_{14} + ib_{14} \\ \times & a_{22} & a_{23} + ib_{23} & a_{24} + ib_{24} \\ \times & \times & a_{33} & a_{34} + ib_{34} \\ \times & \times & \times & \times \end{pmatrix}. \tag{33}$$

All the matrix entries denoted by a "$\times$" are determined due to Hermiticity and tracelessness. We conclude that the representation space is 15-dimensional. The task is now to find 10 linearly independent matrices which are closed under the bracket operation and respect the underlying $su(2) \oplus su(2)$-symmetries. Starting with the latter condition one constructs the following 4×4-matrices using tensor products of the Pauli matrices

$$X^1 = \frac{1}{2}\begin{pmatrix} \sigma_x & \mathbb{0} \\ \mathbb{0} & \mathbb{0} \end{pmatrix}, \qquad X^2 = \frac{1}{2}\begin{pmatrix} \sigma_y & \mathbb{0} \\ \mathbb{0} & \mathbb{0} \end{pmatrix}, \qquad X^3 = \frac{1}{2}\begin{pmatrix} \sigma_z & \mathbb{0} \\ \mathbb{0} & \mathbb{0} \end{pmatrix}, \tag{34}$$

$$Y^1 = \frac{1}{2}\begin{pmatrix} \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \sigma_x \end{pmatrix}, \qquad Y^2 = \frac{1}{2}\begin{pmatrix} \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \sigma_y \end{pmatrix}, \qquad Y^3 = \frac{1}{2}\begin{pmatrix} \mathbb{0} & \mathbb{0} \\ \mathbb{0} & \sigma_z \end{pmatrix}. \tag{35}$$

Here $\mathbb{0}$ is the 2×2 zero-matrix. These matrices yield the following commutation relations:

$$[X^i, X^j] = i\epsilon_{ijk} X^k, \qquad [X^i, Y^j] = 0, \qquad [Y^i, Y^j] = i\epsilon_{ijk} Y^k. \tag{36}$$

Comparing this result with equation (29) we see that we have found a possible representation for the multiplet {4} in $so(4)$ (see figure 5). In order to expand this into so(5), we use the following

[6] A general Lie algebra $SO(n)$ has dimension $n(n-1)/2$.

matrices:

$$K^1 = \frac{1}{2}\begin{pmatrix} \mathbb{0} & i\sigma_x \\ -i\sigma_x & \mathbb{0} \end{pmatrix} \qquad K^2 = \frac{1}{2}\begin{pmatrix} \mathbb{0} & i\sigma_y \\ -i\sigma_y & \mathbb{0} \end{pmatrix} \tag{37}$$

$$K^3 = \frac{1}{2}\begin{pmatrix} \mathbb{0} & i\sigma_z \\ -i\sigma_z & \mathbb{0} \end{pmatrix} \qquad K^4 = \frac{1}{2}\begin{pmatrix} \mathbb{0} & \mathbb{1} \\ \mathbb{1} & \mathbb{0} \end{pmatrix} \tag{38}$$

$$\tag{39}$$

Finally we obtain a set of generators which form a basis:

$$\{X^1, X^2, X^3, Y^1, Y^2, Y^3, K^1, K^2, K^3, K^4\} \tag{40}$$

The remaining commutation relations are listed in appendix 8.1.

3.2 Cartan-Weyl basis

Given a Lie algebra and its commutation relations it is very useful to perform a change of basis such that the Lie algebra can be described in terms of ladder and weight operators. This new basis is called the Cartan-Weyl basis (see subsection 2.3).
Let us introduce new ladder operators:

$$X_\pm = X^1 \pm iX^2, \qquad Y_\pm = Y^1 \pm iY^2,$$
$$U_\pm = K^2 \mp iK^1, \qquad V_\pm = K^4 \mp iK^3.$$

The set of generators is now given by $\{X_+, X_-, X_3, Y_+, Y_-, Y_3, U_+, U_-, V_+, V_-\}$, such that

$$X_+ = (X_-)^\dagger, \qquad Y_+ = (Y_-)^\dagger, \qquad U_+ = (U_-)^\dagger, \qquad V_+ = (V_-)^\dagger. \tag{41}$$

Although we already have a set of ladder and weight operators, we define two new objects which will help us to reach a deeper understanding of $so(5)$,

$$U_3 := X_3 + Y_3, \qquad V_3 := X_3 - Y_3. \tag{42}$$

Since we will work with the resulting commutation relations, they are listed here:

$$
\begin{aligned}
&[X_\pm, X_\mp] = \pm 2X_3, && [Y_\pm, Y_\mp] = \pm 2Y_3, \\
&[X_3, X_\pm] = \pm X_\pm, && [Y_3, Y_\pm] = \pm Y_\pm,
\end{aligned}
$$

$$
\begin{aligned}
&[X_+, U_+] = 0, && [X_-, U_+] = V_-, && [X_3, U_+] = \frac{1}{2}U_+, \\
&[X_+, U_-] = -V_+, && [X_-, U_-] = 0, && [X_3, U_-] = -\frac{1}{2}U_-, \\
&[X_+, V_+] = 0, && [X_-, V_+] = -U_-, && [X_3, V_+] = \frac{1}{2}V_+, \\
&[X_+, V_-] = U_+, && [X_-, V_-] = 0, && [X_3, V_-] = -\frac{1}{2}V_-,
\end{aligned}
$$

$$
\begin{aligned}
&[Y_+, U_+] = 0, && [Y_-, U_+] = -V_+, && [Y_3, U_+] = \frac{1}{2}U_+, \\
&[Y_+, U_-] = V_-, && [Y_-, U_-] = 0, && [Y_3, U_-] = -\frac{1}{2}U_-, \\
&[Y_+, V_+] = -U_+, && [Y_-, V_+] = 0, && [Y_3, V_+] = -\frac{1}{2}V_+, \\
&[Y_+, V_-] = 0, && [Y_-, V_-] = U_-, && [Y_3, V_-] = \frac{1}{2}V_-,
\end{aligned}
$$

$$
\begin{aligned}
&[U_+, U_-] = 2(X_3 + Y_3) =: 2U_3, && [U_+, V_+] = 2X_+, && [U_+, V_-] = -2Y_+, \\
&[V_+, V_-] = 2(X_3 - Y_3) =: 2V_3, && [U_-, V_-] = -2X_-, && [U_-, V_+] = 2Y_-,
\end{aligned}
$$

$$
\begin{aligned}
&[U_3, U_\pm] = \pm U_\pm, && [V_3, Y_\pm] = \pm V_\pm.
\end{aligned}
$$

These commutation relations will be important in order to understand the actions of every operator. We complete our discussion by mentioning once again that with these commutation relations we have completely characterized $so(5)$. We derived them by considering the 4-dimensional spinor representation, but they are also valid for higher-dimensional representations.
The commutation relations for $so(5)$ in the Cartan-Weyl basis imply that we have two weight operators: X_3 and Y_3. So, by definition, $so(5)$ is a rank 2 Lie algebra. As a result multiplets in $so(5)$ can be visualized in a 2-dimensional plane (as we did in figure 6).

3.3 Action of the operators

In the case of $su(2)$ we could transform the basis of generators into a set of ladder and weight operators. There it was easy to see how these operators act on a state $|\Psi\rangle = |m_S\rangle$. However for $so(5)$, since we are talking about a rank 2 Lie algebra, we need to work a bit more:
At this point we have a 10-dimensional basis, which obeys the commutation relations given in section 3.2. Furthermore we defined two objects, U_3 and V_3, which will help us to reach a deeper understanding of the symmetry properties of $so(5)$.
Taking all this into account, one can see that we have eight ladder operators, two "real" weight operators (X_3, Y_3) and two "auxiliary" weight operators (U_3, V_3). Each of the following sets builds up a closed subalgebra equivalent to $su(2)$,

$$\{X_\pm, X_3\}, \qquad \{Y_\pm, Y_3\}, \qquad \{U_\pm, U_3\}, \qquad \{V_\pm, V_3\}.$$

The question we want to answer is which quantum numbers are raised and which ones are lowered, by acting with a certain operator. From the commutation relations we know that

$$[X_3, Y_3] = 0, \tag{43}$$

consequently we can diagonalize both operators simultaneously. We denote the corresponding state with quantum numbers λ_x and λ_y as

$$|\Psi\rangle = |\lambda_x, \lambda_y\rangle\,, \tag{44}$$

such that

$$X_3\,|\lambda_x, \lambda_y\rangle = \lambda_x\,|\lambda_x, \lambda_y\rangle\,, \qquad\qquad Y_3\,|\lambda_x, \lambda_y\rangle = \lambda_y\,|\lambda_x, \lambda_y\rangle\,. \tag{45}$$

This is our starting point. We already know how the operators $\{X_\pm, Y_\pm\}$ act on this state from the case of $su(2)$, e.g., for X_+ we have

$$\begin{aligned}
&[X_3, X_+]\,|\lambda_x, \lambda_y\rangle = X_+\,|\lambda_x, \lambda_y\rangle \\
\Leftrightarrow\; &X_3(X_+\,|\lambda_x, \lambda_y\rangle) = (\lambda_x + 1)(X_+\,|\lambda_x, \lambda_y\rangle),
\end{aligned}$$

$$\begin{aligned}
&[Y_3, X_+]\,|\lambda_x, \lambda_y\rangle = 0\,|\lambda_x, \lambda_y\rangle \\
\Leftrightarrow\; &Y_3(X_+\,|\lambda_x, \lambda_y\rangle) = (\lambda_y)(X_+\,|\lambda_x, \lambda_y\rangle),
\end{aligned}$$

$$\Rightarrow X_+\,|\lambda_x, \lambda_y\rangle \sim |\lambda_x + 1, \lambda_y\rangle\,.$$

The same algorithm holds true for X_-, Y_+, Y_- such that we obtain as expected

$$\begin{aligned}
X_+\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x + 1, \lambda_y\rangle\,, \\
X_-\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x - 1, \lambda_y\rangle\,, \\
Y_+\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x, \lambda_y + 1\rangle\,, \\
Y_-\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x, \lambda_y - 1\rangle\,.
\end{aligned}$$

In the same way one also calculates the actions of U_+, U_-, V_+, V_-. For example, in the case of U_+ one obtains

$$\begin{aligned}
[X_3, U_+]\,|\lambda_x, \lambda_y\rangle &= \frac{1}{2}U_+\,|\lambda_x, \lambda_y\rangle\,, \\
[Y_3, U_+]\,|\lambda_x, \lambda_y\rangle &= \frac{1}{2}U_+\,|\lambda_x, \lambda_y\rangle\,, \\
\Longrightarrow\; U_+\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x + \frac{1}{2}, \lambda_y + \frac{1}{2}\rangle\,.
\end{aligned}$$

We conclude that the actions of U_+, U_-, V_+, V_- are given by

$$\begin{aligned}
U_+\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x + \frac{1}{2}, \lambda_y + \frac{1}{2}\rangle\,, \\
U_-\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x - \frac{1}{2}, \lambda_y - \frac{1}{2}\rangle\,, \\
V_+\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x + \frac{1}{2}, \lambda_y - \frac{1}{2}\rangle\,, \\
V_-\,|\lambda_x, \lambda_y\rangle &\sim |\lambda_x - \frac{1}{2}, \lambda_y + \frac{1}{2}\rangle\,.
\end{aligned}$$

For completeness we also list the actions of U_3 and V_3 which follow by definition

$$U_3\,|\lambda_x, \lambda_y\rangle = (\lambda_x + \lambda_y)\,|\lambda_x, \lambda_y\rangle\,, \qquad\qquad V_3\,|\lambda_x, \lambda_y\rangle = (\lambda_x - \lambda_y)\,|\lambda_x, \lambda_y\rangle\,.$$

3.4 Graphical interpretation of the operators

As in figure 2, we want to represent the actions of the ladder operators graphically. In order to illustrate all ladder operators, the 10-dimensional multiplet $\{10\}$ is convenient, since it is bigger

and gives us more space to clearly arrange everything. Imagine now a state in the middle of the multiplet such that $|\Psi\rangle \sim |0,0\rangle$ (green dot). All other states in the multiplet can be generated by acting on that state with the ladder operators we mentioned before. This also holds true for any lower- or higher-dimensional multiplet and any other starting point in the multiplet. As in the case of $su(2)$, the state we obtain by acting with an operator on a boundary point, leading out of the multiplet, gives zero.

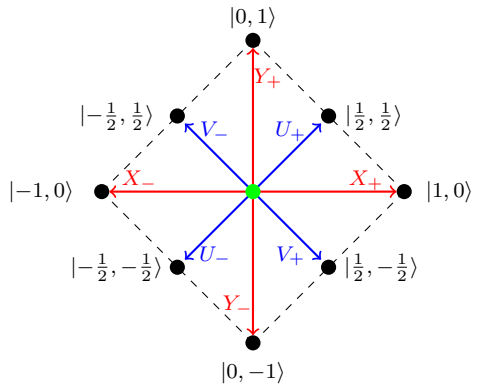


Figure 7: (2,0) = {10} with ladder operators.

3.5 Properties of so(5)

The work we have done so far allows us to discuss symmetry properties of $so(5)$. From the case of $su(2)$ we know that any multiplet is axial symmetric with respect to the origin: if $|S\rangle$ is an available state so there is $|-S\rangle$ (See figure 2). This is known as a $\mathbb{Z}_2$-symmetry.
The four weight operators X_3, Y_3, U_3, V_3, commute with each other, meaning that they can be diagonalized simultaneously. Their corresponding $su(2)$-subalgebras

$$\{X_\pm, X_3\}, \qquad \{Y_\pm, Y_3\}, \qquad \{U_\pm, U_3\}, \qquad \{V_\pm, V_3\},$$

are closed. Hence we can impose four symmetry axes for an arbitrary $so(5)$ multiplet:

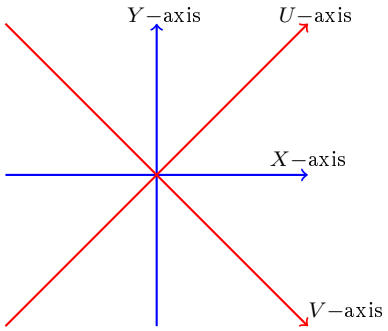
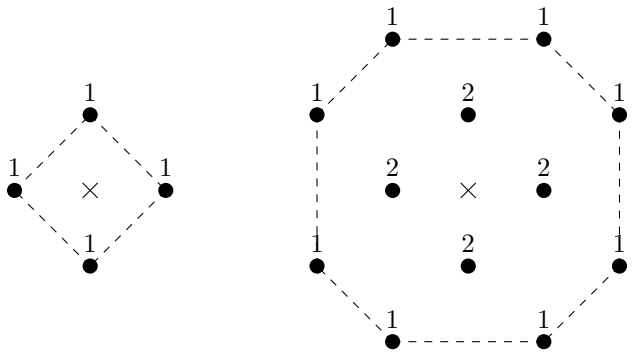


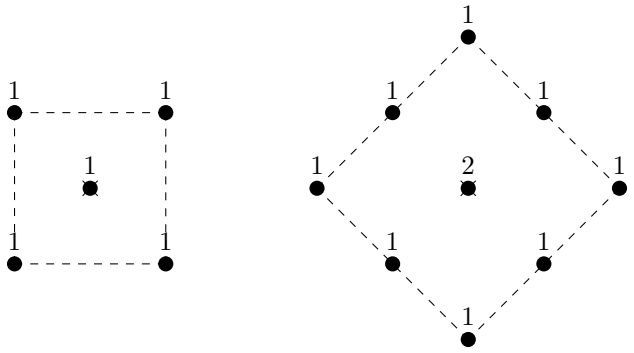
Figure 8: Symmetry axes for $so(5)$.

Those symmetry axes are characteristic for the Lie algebra $so(5)$. As a result, any multiplet in $so(5)$ has the shape of an **octagon**, which is characterized by its side lengths q along the X, Y axes and p along the U, V axes. We have already encountered two examples obeying these rules: The spinor representation $(p,q) = (1,0) = \{4\}$ in figure 6 and the adjoint representation $(2,0) = \{10\}$ in figure 7. Here we show some non-trivial low-dimensional irreducible multiplets of $so(5)$.



$(1,0) = \{4\}$ $(1,1) = \{16\}$

Figure 9: Multiplets with non-trivial duality.



$(0,1) = \{5\}$ $(2,0) = \{10\}$

Figure 10: Multiplets with trivial duality.

The dimension of a multiplet, i.e., its total number of states, is determined by p and q and given by (see [4])

$$D(p,q) = \frac{1}{6}(p+1)(q+1)(p+q+2)(p+2q+3), \tag{46}$$

such that for a given p and q we have (being consistent with our previous notation)

$$(p,q) = \{D(p,q)\}. \tag{47}$$

As already mentioned, a multiplet is completely characterized by p and q and not only by its dimension, for instance

$$(2,1) = (4,0) = \{35\}. \tag{48}$$

As a result we can find two sets of 35-dimensional matrices which obey the commutations relations for $so(5)$. In order to distinguish the corresponding multiplets we denote

$$(2,1) = \{35\}, \qquad (4,0) = \{35^*\}. \tag{49}$$

The same procedure also applies for all other degeneracies for $D(p,q)$.

3.5.1 Multiplicities for so(5)

In contrast to the Lie algebra $su(2)$, some states in an $so(5)$ multiplet may be degenerate, i.e., there may be several states having the same eigenvalues λ_x,λ_y. The number of different states

for given λ_x,λ_y in a representation is called multiplicity. In $su(2)$ all states have multiplicity one. Starting from the highest weight state $|\lambda_{max}\rangle$ (which has multiplicity one), there is only one way to obtain all other states: by acting iteratively with the lowering operator S_- (see figure 2). Consider the adjoint representation $\{10\}$ given in figure 10. Starting from the highest weight state $|1,0\rangle$ with multiplicity 1, one can see that there exist two independent ways of getting to the point $|0,0\rangle$.

$$\begin{aligned} |0,0\rangle &\sim X_- \,|1,0\rangle\,, \\ &\sim (U_-V_-)\,|1,0\rangle\,, \end{aligned}$$

which results in two independent states with eigenvalues $\lambda_x = 0, \lambda_y = 0$. Using the way V_-U_- doesn't give any new states since

$$(V_-U_-)\,|1,0\rangle = ([V_-,U_-] + U_-V_-)\,|1,0\rangle = (2X_- + U_-V_-)\,|1,0\rangle\,.$$

We conclude that the state with eigenvalues $\lambda_x = 0, \lambda_y = 0$ is two-fold degenerated, i.e., it has multiplicity 2. Unfortunately, there is no easy pattern of calculating these multiplicites for higher-dimensional representations; the method to consider different ways into the middle fails at higher-dimensional multiplets. Let us consider an 81-dimensional representation $(p,q) = (2,2) = \{81\}$ which results in the following multiplet:

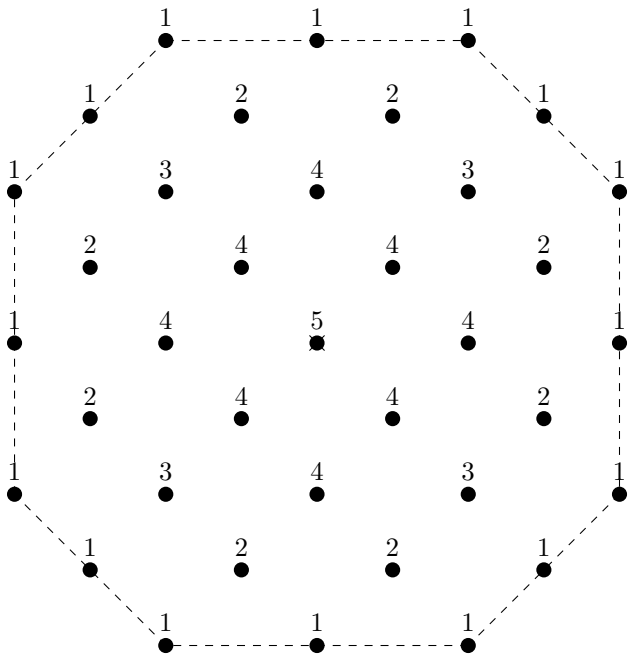


Figure 11: The representation $\{81\}$ in $so(5)$.

One can see that the multiplicities for a given pair of quantum numbers $|\lambda_x, \lambda_y\rangle$ agrees with their symmetry partners, i.e., if $|\lambda_x, \lambda_y\rangle$ has multiplicity m, so do all other points in the multiplet that one gets by mirroring on the symmetry axes X, Y, U, V. The multiplicities on the boundary are always 1, since they are highest weight states. The multiplicities in the inner part of the multiplet increase (or stay the same). Taking a closer look, one sees that the multiplicities do not follow any obvious pattern. For certain cases one finds regularities but often they do not hold true for higher-dimensional representations. In order to determine those multiplicities one can use the so-called **recursive Freudenthal-Formula** (see[7], page 444). Although the formula of Freudenthal is quite intuitive, it requires a lot more theory and, in addition, it is also tedious to use by hand. However, the scheme described by Antoine and Speiser provides a tool to calculate multiplicities in a rather easy way.

4 Tensor product reduction

So far we have encountered Lie algebras, representations, and their multiplets. Now we want to discuss the concept of coupling two representations, respectively their multiplets. An algebraic variant of reducing tensor products is given by the so-called *girdle-method* ([6]). For the algebras $so(n)$, $su(n)$ and $sp(n)$ there are useful schemes based on *Young-tableaux* ([7]). Since these methods are tedious and not very intuitive, we will not discuss them here. Instead, we discuss a graphical tensor product reduction scheme discussed by Greiner (also ([7])) and its disadvantages compared to the reduction scheme described by Antoine and Speiser. In order to keep it simple and evident, we neglect strict mathematical descriptions and give graphical interpretations of these methods.
In the introduction we coupled two $su(2)$ multiplets: $\{2\}$ and $\{5\}$ such that

$$\{2\} \otimes \{5\} = \{4\} \oplus \{6\}.$$

In this particular case one could use the well-known coupling rules which are used to couple angular momenta. The tensor product of two representations with spin S^1 and spin S^2 results in all representations with total spin between $|S^1 - S^2|$ and $S^1 + S^2$:

$$\{2S^1 + 1\} \otimes \{2S^2 + 1\} = \{2|S^1 - S^2| + 1\} \oplus \cdots \oplus \{2(S^1 + S^2) + 1\}.$$

This coupling rule for $su(2)$ is somewhat arbitrary and seems to appear from nowhere. In order to understand the coupling of two representations in $so(5)$ we need to rethink this problem in a more abstract way.

4.1 Interpretation of coupling two representations

Coupling two representations $\{2\}$ and $\{5\}$ in $su(2)$ means that we couple every state of $\{5\}$ with every state of $\{2\}$ (or vice-versa). More precisely we take every state of $\{5\} = \{|-2\rangle, |-1\rangle, |0\rangle, |1\rangle, |2\rangle\}$ and "shift" them by the states of $\{2\} = \{|-\frac{1}{2}\rangle, |\frac{1}{2}\rangle\}$. This procedure can be represented graphically by superimposing the multiplet $\{2\}$ one every state of the multiplet $\{5\}$, such that we obtain all possible states in the direct product. The resulting multiplet can be decomposed into two irreducible ones.

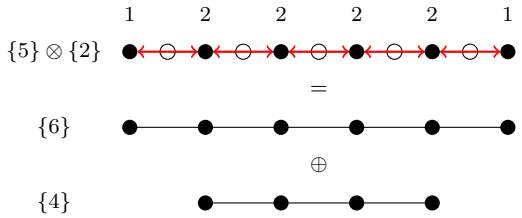


Figure 12: Coupling $\{5\}$ with $\{2\}$

This method can be described in the following general way: Given two multiplets we want to couple, we can superimpose the second multiplet on every state of the first one. The resulting multiplet represents all possible obtainable states in the direct product and decomposes into a direct sum of irreducible multiplets.

4.2 Coupling two representations in so(5)

This concept of calculating tensor products also generalizes to rank 2 tensor products. Since we are working with $so(5)$, we want to do our first tensor product reduction using this particular method: So let us couple the two fundamental non-trivial representations $\{4\}$ and $\{5\}$ of $so(5)$, by superimposing $\{4\}$ on every state of $\{5\}$[7].

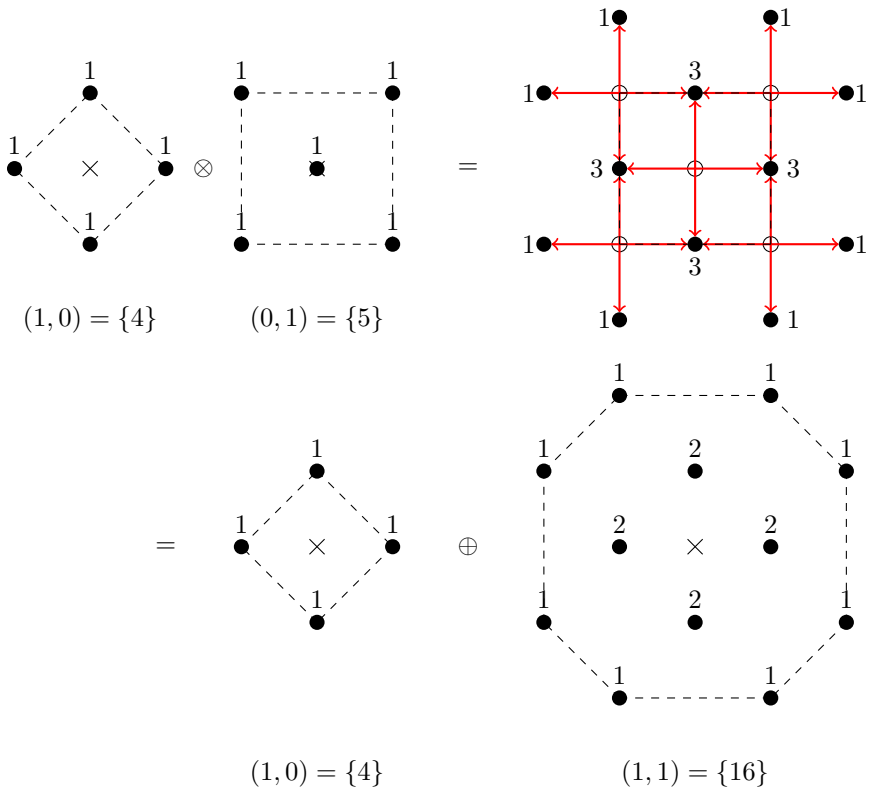


Figure 13: Coupling: $\{4\} \otimes \{5\} = \{16\} \oplus \{4\}$.

We finally performed our first tensor product reduction of two irreducible multiplets with the result that

$$\{4\} \otimes \{5\} = \{4\} \oplus \{16\}.$$

Using this example we have just seen that the calculation is quite intuitive and can be carried out with simple graphical considerations. However, already in this case the calculation was cumbersome since we had to make lots of drawing; assume we wanted to calculate $\{10\} \otimes \{10000\}$ (see equation (3)). One can easily see that the effort to calculate this by hand is enormous. A computer on the other hand, could do these calculations quicker, nevertheless the working time for the computer increases drastically for higher-dimensional multiplets.
At this point the so-called *landscape* introduced by Antoine and Speiser comes into play. The landscape allows us to calculate tensor products using simple additions and subtractions.

[7] Note: we are coupling the ones in figure 9 and figure 10.

5 Antoine-Speiser tensor product reduction scheme

As discussed by Antoine and Speiser ([3, 4]), the multiplet of a rank 2 Lie algebra can be positioned in a two-dimensional landscape. This landscape allows us to calculate multiplicities and tensor products of irreducible representations.

5.1 Landscape of su(2)

First we want to work out the landscape for $su(2)$. We know that $su(2)$ is a rank 1 Lie algebra, so the landscape is one-dimensional, i.e., it extends along one line. Imposing that all irreducible representations of $su(2)$ need to be situated on this line, it is quite intuitive to see how the landscape should look like.

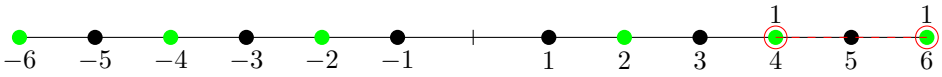


Figure 14: landscape for $su(2)$, $\{2\}$ (red) superimposed on $\{5\}$.

The landscape of $su(2)$ is divided into a positive sector on the right and a negative sector on the left. It consists of two sublattices one for the integer (black dots) and one for the half-integer (green dots) representations, due to the center of the covering group (see subsection 2.8). Coupling once again the two representations $\{5\}$ and $\{2\}$ in $su(2)$ using the landscape means that we superimpose $\{2\}$ on the dot in the landscape which belongs to the irreducible representation of $\{5\}$. As a result we immediately obtain our well-known result

$$\{2\} \otimes \{5\} = \{4\} \oplus \{6\}.$$

Of course one could also superimpose $\{5\}$ on $\{2\}$. Naturally we get the same result

$$\{2\} \otimes \{5\} = \ominus\{2\} \oplus \{2\} \oplus \{4\} \oplus \{6\} = \{4\} \oplus \{6\}.$$

It is quite evident that one can save time by superimposing the lower-dimensional multiplet on the higher-dimensional one. One can also see that the tensor product reduction in $su(2)$ is rather simple and intuitive: What we basically did, is to apply the coupling rules for angular momenta graphically. However, given the simple algebraic coupling rules for angular momenta, $su(2)$ does not really require a graphical tensor product reduction method.

5.2 Landscape of so(4)

The concept of the landscape develops its full strength for rank 2 Lie algebras, such that we first want to consider the simple case Lie Algebra $so(4) = su(2)_x \oplus su(2)_y$. Since $so(4)$ is just the sum of two independently acting $su(2)$ subalgebras all states are non-degenerate (i.e. have multiplicity one). The landscape can be built by using two times the already known $su(2)$-landscape.
One can easily show, that each multiplet in $so(4)$ has the shape of a rectangle, determined by its height h and its side length w. In this particular case it is vital to classify the multiplets in terms of their $su(2)$ subalgebras. For instance the resulting multiplet from the vector representation (see figure 5) can be written as

$$\{4\} = \{2, 2\},$$

which takes into account that $\{4\}$ is a doublet with respect to $su(2)_x$ and to $su(2)_y$. Due to the symmetry properties of $so(4)$ we have degeneracies with respect to the dimension of the multiplets. For instance:

$$\{2, 3\} = \{6\}, \qquad \{3, 2\} = \{6^*\}.$$

The multiplet $\{2, 3\}$ is a doublet with respect to $su(2)_x$ and a triplet with respect to $su(2)_y$. Hence, this representation has trivial duality with respect to $su(2)_y$. For the case of $\{3, 2\}$ the

inverse holds true.
This fact is also taken into account in the landscape. Each of the points in the landscape represents an irreducible representation of $so(4)$. As we stated in subsection 2.8, $so(4)$ is a special case: we distinguish between 4 different types of representations depending on the duality with respect to the subalgebras $su(2)_x$ and $su(2)_y$. The black dots are representations of trivial duality with respect to both subalgebras. The blue dots are representations with trivial duality with respect to $su(2)_x$ and non-trivial with respect to $su(2)_y$, purple dots behave vice-versa. The green dots are representations of non-trivial duality with respect to both $su(2)$ subalgebras. As a result $\{3,2\}$ is represented in the landscape by a blue dot, $\{2,3\}$ by a purple one and the vector representation $\{4\}$ is denoted by a green dot.
Let us now couple two representations of $so(4)$ using the landscape, for example $\{4\}$ and $\{6^*\}$. We do this, as in the case of $su(2)$, by superimposing the multiplet of $\{6^*\}$ on the representation in the landscape, which belongs to $\{4\}$.

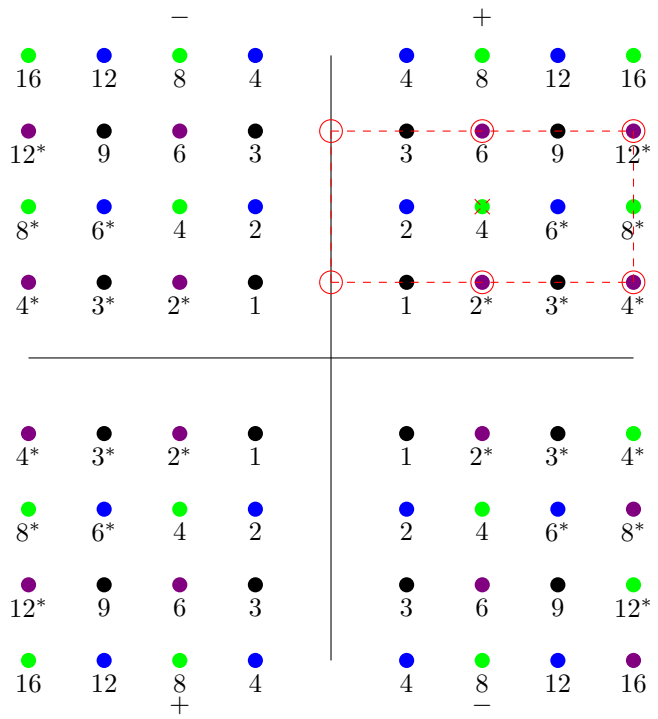


Figure 15: Central part of the landscape $so(4)$, $\{6^*\}$ superimposed on $\{4\}$.

Using the graphical vector method described in 4.2 would give us a large amount of work; using the landscape we easily get the result

$$\{4\} \otimes \{6^*\} = \{12^*\} \oplus \{6\} \oplus \{4^*\} \oplus \{2^*\}.$$

5.3 Landscape of so(5)

Due to the properties of $so(5)$, its landscape has a unique form that distinguishes it from other rank 2 Lie algebras. Up to now we constructed landscapes just by using multiples of $su(2)$-landscapes. However, for $so(5)$ the situation is somewhat more complicated: since $su(2)_x \oplus su(2)_y = so(4) \subset so(5)$ we still have two $su(2)$ subalgebras which could be used in order to classify multiplets. Due to the interaction between those $su(2)$ algebras, namely represented by the ladder operators $U_\pm, V_\pm$, the subalgebras do not act independently and so we do have to take into account multiplicities. At this point we make use of the $so(5)$-landscape provided by Antoine and Speiser ([3, 4]) and do not derive it. Here we provide the central sector of the landscape, a bigger one can be found in the appendix (8.2).

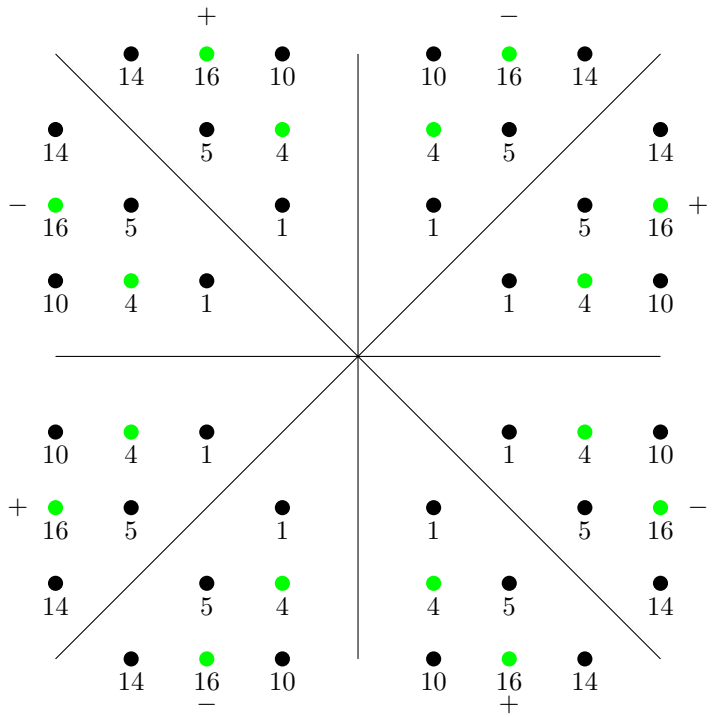


Figure 16: Central part of the landscape of $so(5)$.

The landscape is divided into eight segments with alternating sign. Any dot in the landscape represents an irreducible representation of $so(5)$. The representations of trivial and non-trivial dualities are denoted by different colors: The ones with trivial duality are black, the ones with non-trivial duality are green. The Cartesian co-ordinates of any representation $(p, q) = \{D(p, q)\}$ in the landscape are given by

$$x = p + q + 2, \qquad y = q + 1, \tag{50}$$

such that the dimension reads

$$D(x, y) = \frac{1}{6}(p + 1)(q + 1)(p + q + 2)(p + 2q + 3) = \frac{1}{6}xy(x^2 - y^2). \tag{51}$$

As in $so(4)$, we have cases where the dimension is degenerate. In this case the point in the landscape is denoted with an additional "*", e.g., the representation $\{35^*\}$ in the landscape is also denoted with the additional "*". It is worth noticing that the landscape contains all symmetry properties of $so(5)$, as discussed in 3.5.

5.4 Coupling two representations in so(5)

Since we are given the landscape by J. P. Antoine and D. Speiser, we are prepared to discuss the tensor product reduction scheme. Given the landscape of $so(5)$ we can couple any two multiplets like we did in the case of $so(4)$, described in subsection 5.2. However, in the case of $so(4)$ all states had multiplicity one: this does not hold true for $so(5)$. First of all we need an algorithm for calculating the multiplicities for a given multiplet, using the $so(5)$-landscape.

5.4.1 Multiplicities for so(5) multiplets

The multiplicities for a given multiplet only depend on (p, q): The side-length along the diagonal and the side-length along the horizontal. In order to determine the multiplicities for a given multiplet $(p, q) = \{D\}$, we couple $\{D\}$ with the singlet $(0, 0) = \{1\}$ in the landscape, such that

$$\{D\} \otimes \{1\} = \{D\}. \tag{52}$$

It is easy to see that this equation holds true, if we remember the physical interpretation of coupling two multiplets (see subsection 3.4). We superimpose $\{D\}$ on the singlet $\{1\}$ and see that a number of representations in the landscape are covered by states of the multiplet. The contribution we want (i.e. the point in the landscape which corresponds to the multiplet $\{D\}$) is situated in the upper-right boundary of the multiplet, in the positive sector, and nowhere else. As a result this state has multiplicity one, which coincides with the fact that it is a highest weight state. All other contributions (i.e. all other points covered by the multiplet) need to sum up to zero. This is the condition needed such that equation (52) holds true. Since we know one state has multiplicity 1, all symmetry partners, i.e., all states one gets by mirroring on the X, Y, U, V axes (see subsection 3.5), also have multiplicity 1. Using the 8-fold symmetry and the fact that all other contributions (except for the one given by $\{D\}$) need to sum up to zero, one can easily determine all multiplicities recursively, starting from the point which covers $(p, q) = \{D\}$.
Let us calculate the multiplicites of $(1, 1) = \{16\}$; a representation we already met in figure 9, and let us assume we do not know the multiplicites yet. Using the described algorithm we get:

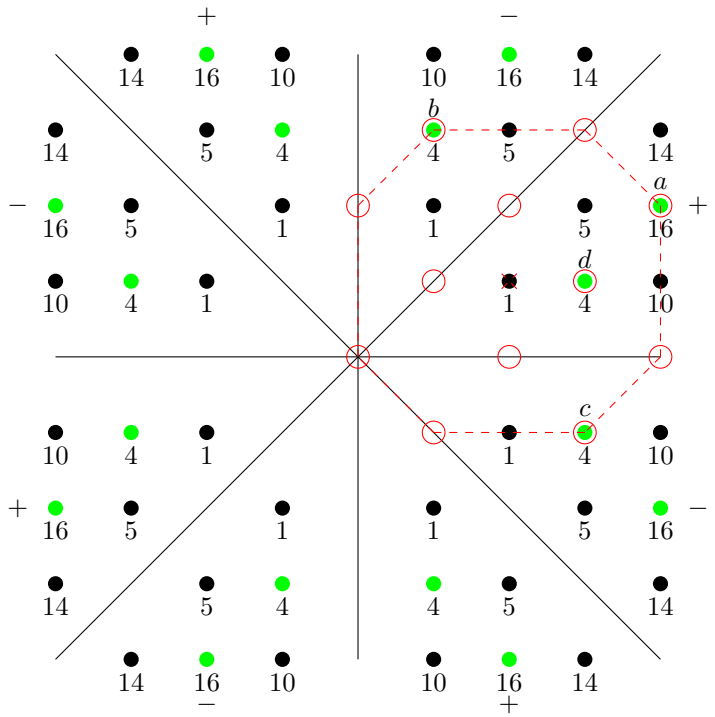


Figure 17: Calculating multiplicities for $(1, 1) = \{16\}$.

We see that we get the following contributions such that

$$\{16\} \otimes \{1\} = a\{16\} + (d-b-c)\{4\} \stackrel{!}{=} \{16\}. \tag{53}$$

Symmetry-considerations give us $a = b = c = 1$ and $d = 2$.

5.4.2 Coupling two representations in so(5)

At this point we have all the tools needed in order to perform our first tensor product reduction using the landscape. In subsection 4.2 we coupled the 4-dimensional spinor representation with the 5-dimensional vector representation using an intuitive graphical method described in Greiner's "Symmetries" ([7]). Let us now check if

$$\{4\} \otimes \{5\} = \{4\} \oplus \{16\}$$

is also reproduced using the Antoine and Speiser reduction method. First we have to determine the multiplicities by superimposing $\{4\}$ and $\{5\}$ on the singlet $\{1\}$ in the landscape. It can easily be seen that in both representations for every pair of quantum numbers λ_x, λ_y we have multiplicity 1, as we already stated indirectly in figure 9 and figure 10. Since we have determined the multiplicities of both representations, we can finally couple them by superimposing the multiplet $\{4\}$ on the point in the landscape which belongs to $\{5\}$. Reading off the covered representation in the landscape leads to the correct result we just mentioned before.

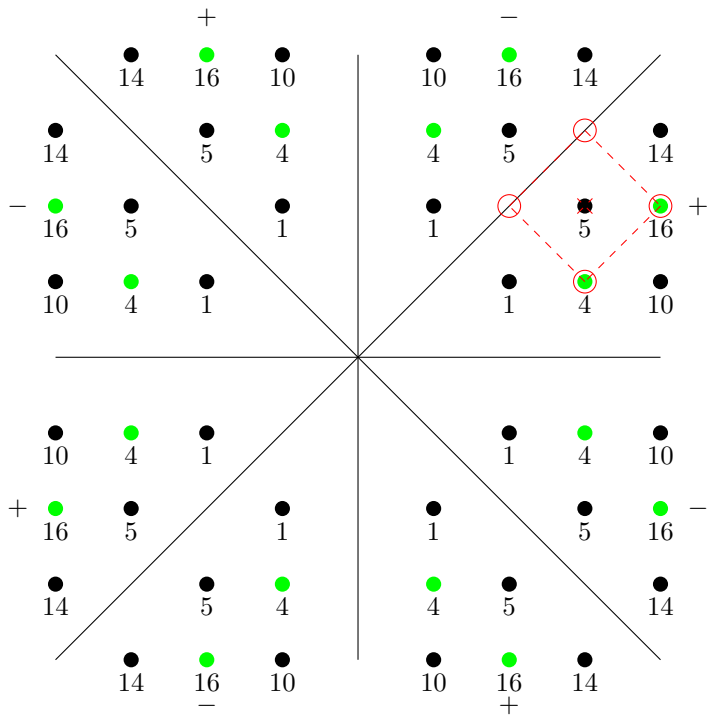


Figure 18: Coupling $\{4\} \otimes \{5\}$ in $so(5)$.

We immediately see that this method using the landscape is much more efficient than using the graphical vector-method described by Greiner. Of course, one can couple any desired multiplets; even very high-dimensional ones are an easy task now. For example, one could now easily calculate $\{10000\} \otimes \{10\}$ by just superimposing $\{10\}$ on the landscape and reading off the 10 representations which are covered by the multiplet.

5.5 Final comment

As we have seen so far, calculating tensor products can be a very tedious and time-consuming problem. Having appropriate tools, in order to perform these calculations, is very important and time-saving. The challenging task is to write a program which "understands" how the tensor product reduction scheme works and also executes it. In this particular case the Antoine and Speiser tensor product reduction scheme for $so(5)$ was implemented into a Java-program which we want to present now in a general manner, without going into all fine details.

6 Antoine and Speiser-scheme implemented into a Java-program

6.1 Introduction to the Java-language

As we stated at the beginning of this thesis the program was written in Java-language. Here we provide basic informations on how a generic Java-program works, before considering the written program itself. Java is a so-called *object-oriented language*, meaning that *objects* are the fundamental entities in a Java-program. A Java-object often represents a real entity needed in order to solve a problem. An object is defined by a *class*. A class is the model or blueprint from which the object is created. Multiple objects can be created from one class definition. Inside the class one defines *states*, potential *attributes* and behaviours an object can have. By *state* we mean "state of being" - fundamental characteristics that define the object. An object's *attributes* are the values an object stores internally, e.g., height, color and so on. *Methods* describe the potential behaviour of an object; they are a group of programming statements given a name. When a method is invoked, its statements are executed.
As an example one could think of a class called "car". The car is defined by its "state": A car has four wheels, an engine and so on. Attributes could be the brand and horsepower. A method we want to execute having an car-object would be "drive to university".
Having all this, we can define a car-object: Let us call it "Tim's car", which has all characteristics a car needs. We define the car to be a Mercedes and want it to have 150 horsepower. Having that object defined, we can invoke the method called "drive to university": and then drive to university.
Since the structure of the program is not complicated in the sense that it requires the whole repertoire of all Java-functions, the previously presented notions are all we need in order to understand the program.

Graphical overview of the program

At this point we have all the terms needed in order to illustrate the program in a general way. First of all we give a graphical overview before considering the main parts of the program.

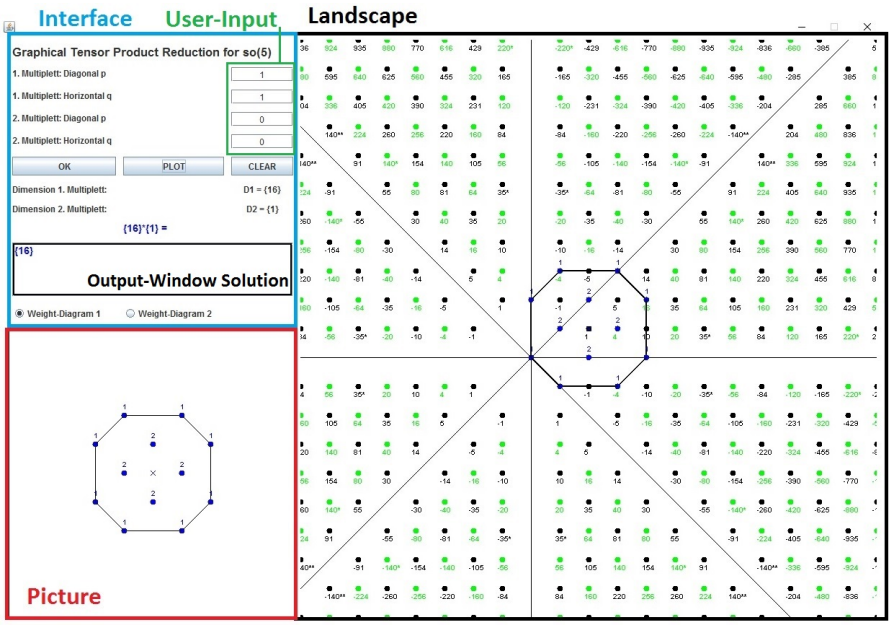


Figure 19: Graphical overview program.

6.2 Structure of the program

As already mentioned the program is divided into *classes* and their corresponding objects. The graphical overview shows us the frame we see, when executing the program. In order to understand what happens behind the window, we also want to give an overview of the different classes and briefly explain what they do. For the sake of clearness we are sometimes going to denote the **object** with the same name as the **class**, in the case where multiple definitions would just cause confusion.

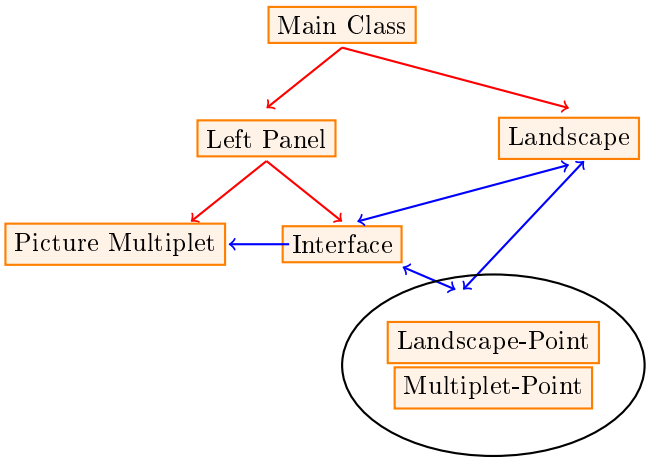


Figure 20: Overview of the different classes.

6.2.1 Main Class

The main class is the superior class which contains the execution statement. It has the following properties:

- Contains the execution command.
- Sets up the frame (creates the Graphical User Interface GUI).
- Defines an object **Left Panel** from the class *Left Panel*.
- Defines an object **Landscape** from the class *Landscape*.
- Sorts both objects in the frame as in figure 19.

6.2.2 Left Panel

The Left Panel is a container which describes the left-hand side of the frame. The following tasks are executed:

- Defines an object **Interface** from the class *Interface*.
- Defines an object **Picture** from the class *Picture*.

6.2.3 Landscape

Before taking a look at the main algorithm, we want to give some details about the object Landscape. The Landscape was the first object programmed in this problem statement, such that it also contains features which work independently from all other components of the program. The Landscape defines four two-dimensional arrays, for each quadrant of the $so(5)$-landscape. Each entry of these arrays is an object from the class *Landscape-Point*, which denotes a representation in the Landscape. Each representation is printed into the landscape. Representations which are

degenerate, due to the dimension $\{D(p,q)\}$ are denoted in the landscape with an additional "*". Furthermore the landscape has the following two features which work independently from the rest of the program:

- "Drag and Drop"- function, such that one can move inside the landscape.
- If needed one can also "delete" representations by pressing the *right mouse button* on the representation in the landscape. Pressing once again results in the representation to re-appear.

The command in order to execute the Antoine and Speiser tensor product reduction scheme comes from the object **Interface**. Although the landscape could work independently for itself, it also waits until someone gives a User-Input in the interface and presses the "**PLOT**"-button. Having an input the landscape superimposes the desired multiplet in the landscape, reads off which points are covered and sends the information back to the interface. We are going to specify this idea, when considering the interaction between the objects **Interface** and **Landscape**.

6.3 Interface and Picture

The interface is the "heart" of the program: it creates multiplets, based on the input of the user and calculates multiplicities by creating a landscape which runs in the background. This "imaginary" landscape calculates the multiplicities by executing the algorithm given in subsubsection 5.4.1. Furthermore the interface sets up the interaction between itself and the "real" landscape on the right hand side of the frame. We want to summarize the **Interface**'s actions with the following list of statements it executes:

- Creates the Graphical User Interface (sets up buttons, labels, text fields and so on), connects those buttons with actions that are executed.
- Creates a set of objects from the class *Landscape-Point*, i.e., it creates an array, where all the representations of the landscape are listed (including their dimension, co-ordinates, degeneracies etc.). One can think of a complete landscape running in the background.
- When pressing "**OK**" the interface constructs two 2-dimensional arrays, whose elements belong to the class *Multiplet-Point*, depending on the User-Input. The algorithm, by which those multiplets are completely characterized, is described in subsection 6.4.
- The finished multiplets are handed over to the object **Picture**, where one of the multiplets is displayed, depending on the selected Radio-Button (*Weight-Diagram 1* or *Weight Diagram 2*).
- If "**PLOT**" is pressed the program executes the same algorithm as when "**OK**" is pressed. Additionally the program superimposes the desired multiplet on the landscape and calculates its contributions.
- When pressing "**CLEAR**", all settings are reset.

6.4 Construction algorithm for a multiplet

Once the user gives in the side-lengths for both multiplets and presses "**OK**" (or "**PLOT**") the program determines the multiplets based on a tedious algorithm. The creation of multiplets - especially determing the multiplicities - is quite complicated, such that we want to give an idea of the algorithm used. In order to create a multiplet only two inputs are needed: the side-length along the diagonal p and the side-length along the horizontal q. So let us assume we get the input for the first multiplet: $p1$ and $q1$. The interface executes the following algorithm for the first multiplet[8]:

[8] For the second multiplet the algorithm is the same. Since we perform the algorithm for the first multiplet, every variable has an additional "1" at the end.

1. We define the integers

$$numshell1 = (int)(\frac{p1 + 2q1}{2}) + 1,$$
$$maxnum1 = (int)4(p1 + q1).$$

The integer $numshell1$ determines the number of "shells" a multiplet has, and is rounded down if needed. The number $maxnum1$ determines the number of multiplet points one can find in the outer shell of the multiplet.
If $p1 = q1 = 0$, the program simply defines $numshell1 = maxnum1 = 1$, such that we get the singlet $\{1\}$.

2. A two-dimensional array, which we call $multiplet1$, is created. The number of entries is based on the two previously defined variables:
$multiplet1 = muliplet1[numshell1][maxnum1]$.

3. Each entry of this array is an object from the class *Multiplet-Point*, which has attributes like position and multiplicity.

4. In order to hand in co-ordinates to every point of the multiplet (i.e. to every entry of the array), we start by the point in the lower-left part of the multiplet. The position of this particular point is defined to be:

$$x = -(int)(\frac{l_q}{2})(p1 + q1),$$
$$y = (int)(\frac{l_q}{2})q1,$$

where l_q is the standard parameter of length. Based on $p1$ and $q1$ the program calculates the positions of the outer shell clockwise. When the starting point is reached, the algorithm proceeds to the inner part of the shell structure and so on.

At this point we want to give an example: Consider the multiplet based on $p1 = 1, q1 = 1$. Simple calculations yield

$$numshell1 = \lceil 2.5 \rceil = 2, \qquad maxnum1 = 8.$$

The array $multiplet1[2][8]$ is created and any entry in the array is associated with a point of the multiplet:

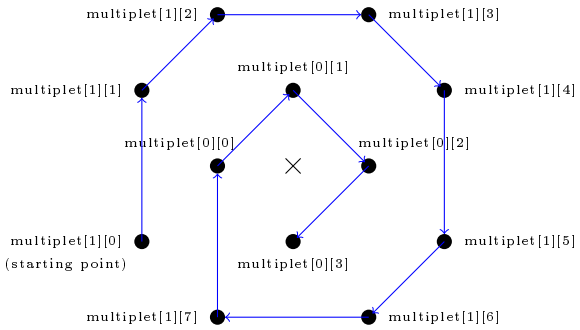


Figure 21: Generating the multiplet $(1, 1) = \{16\}$.

Having calculated the co-ordinates for every point of the multiplet, we need to calculate the multiplicities.

5. In order to calculate multiplicities we use the previously mentioned class *Landscape-Point*. Analogously, as for the landscape, we define four two-dimensional arrays where each entry

denotes a representation in the landscape, with its co-ordinates. The co-ordinates of this "imaginary" landscape are now shifted in such a way, that the singlet $\{1\}$, in the positive sector on the right, is shifted to the middle. The program superimposes the given multiplet on the representation which belongs to $\{1\}$ and calculates the co-ordinates which are covered by the multiplet. As a result, some **multiplet-points** are given an internal value others not.
As a side note we also want to discuss another interesting property of the program, namely the way how the program calculates the dimension $\{D(p,q)\}$ of a multiplet.
Superimposing the multiplet with the singlet, one can uniquely determine the dimension of the multiplet. Using the dimension formula would not take into account the degeneracies of the dimensions. However, the landscape does take degeneracies into account (since it was programmed that way). The $2(p1+q1)$' entry of the first shell (respectively the entry $multiplet1[numshell1-1][2(p1+q1)]$) always covers the correct dimension of the multiplet (see figure 17).

6. Next the program executes a method which divides all points of the multiplet into groups of symmetry partners. These symmetry partners are given the same value of m: the multiplicity. The sets of symmetry partners are then enumerated: $m_1, m_2, \ldots$.

7. As a result we obtain a set of equations such that equation (52) needs to be fulfilled. These equations are solved by an external program called **Jama** ([8]), by writing these equations into matrix form and inverting it.

8. Finally the calculated multiplicities are given back to the symmetry partners and therefore to the multiplet-points. We obtain a completely characterized multiplet.

Let us return to our example. As in subsection 5.4.1, the program splits the set of multiplet-points into groups of symmetry partners: here specifically denoted by different colours. In addition to this, some entries of $multiplet1$ are given a value: namely the value obtained by superimposing the multiplet with the singlet. The different groups of symmetry partners are given an index: As a result we get a set of equations which is put into a matrix

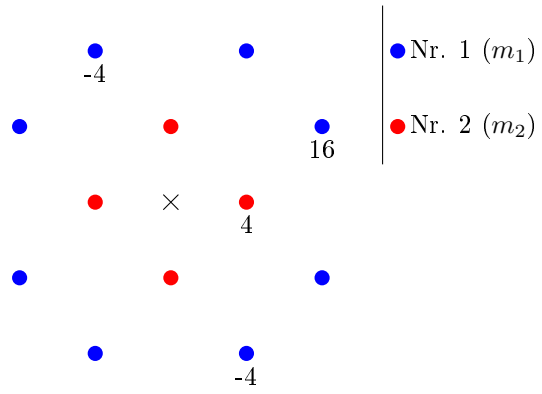


Figure 22: $(1,1) = \{16\}$, with the contributions we get by coupling $\{16\} \otimes \{1\}$.

$$\begin{aligned} 1 \cdot m_1 &= 1, \\ -2 \cdot m_1 + 1 \cdot m_2 &= 0, \\ &\Leftrightarrow \\ \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \end{pmatrix} &= \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \end{aligned}$$

The program **Jama** solves this matrix equation and gives us the well-known result: $m_1 = 1$, $m_2 = 2$.

6.5 Interaction between the interface and the landscape

Up to now we didn't interact with the landscape on the right-hand side of the frame and basically just described the **Left Panel** of the program. However, when pressing the "**PLOT**"-button interaction between the object **Interface** and **Landscape** is invoked. The constructed multiplets in subsection 6.4 are handed over to the "real" landscape. Depending on the radio-button selected (*Weight-Diagram 1* or *Weight-Diagram 2*) either the first multiplet is superimposed at the landscape-point of the second multiplet, or vice-versa.
Let us assume we superimpose the first multiplet on the point in the landscape of the second one. First the landscape is shifted until the dot for the second multiplet is centered. Then the first multiplet is superimposed and the object **Landscape** calculates all the intersections between the **Landscape-Points** and the **Multiplet-Points**.
The landscape also calculates all contributions, summarizes and sorts them. The result is sent back to the interface where it is printed in a *JScrollPanel* (A text-field with scrolling option).
As already mentioned the "**CLEAR**"-button of the interface resets all settings and puts them back into default-mode. The multiplets in the object **Picture** and **Landscape** disappear.

7 Conclusion and outlook

The aim of this thesis was to get an overlook about Lie algebras -especially about $so(5)$-, and to implement the tensor product reduction scheme, described by J. P. Antoine and D. Speiser. We started with the concept of a *Lie group* and continued by considering the tangential space of the group, where we obtained the generators of the corresponding *Lie algebra*. As the commutations relations were fixed, we started considering *representations*, *multiplets* and their *multiplicities*. Having the concept of *multiplets* we coupled several multiplets by using a cumbersome graphical vector method. Then we introduced the *landscape* and presented a much more economical way of calculating tensor products, namely the tensor product reduction method by Antoine and Speiser. Finally we gave a brief overlook, about the Java-program, where this particular method has been implemented.
The geometry of Lie algebras is very interesting and important in order to describe symmetries in nature, but it is also worth mentioning the beauty of such objects.
We conclude our discussion by mentioning that in this thesis we only considered a special type of Lie algebras, namely rank 1 and rank 2 Lie algebras. One can immediately imagine that there could be some sort of "three-dimensional landscape" where all representations of a given rank 3 Lie algebras live[9].

7.1 Acknowledgement

A special thank goes to Prof. Uwe-Jens Wiese for the support and interesting conversations. I would also like to thank Felix von Rütte, "my precursor", for his suggestion to use the Java-language in order to implement the tensor product reduction scheme. Another thank goes to all my fellow students who tested the program, and found out programming errors.
Throughout this thesis I talked to many people: Professors, fellow students and family members. Everyone of them was helpful and I would like to thank everyone who directly or indirectly helped me throughout this thesis.

[9] rank 3 Lie algebras are, for example, $su(4) \simeq so(6), sp(3)$.

8 Appendix

8.1 Appendix A: Commutation relations primitive basis

Here we list the commutation relations for the generic set of generators of $so(5)$.

$$
\begin{array}{lll}
[X^i, X^j] = i\epsilon_{ijk} X^k & [X^i, Y^j] = 0 & [Y^i, Y^j] = i\epsilon_{ijk} Y^k \\
[X^1, K^1] = \frac{i}{2} K^4, & [X^2, K^1] = -\frac{i}{2} K^3, & [X^3, K^1] = \frac{i}{2} K^2, \\
[X^1, K^2] = \frac{i}{2} K^3, & [X^2, K^2] = \frac{i}{2} K^4, & [X^3, K^2] = -\frac{i}{2} K^1, \\
[X^1, K^3] = -\frac{i}{2} K^2, & [X^2, K^3] = \frac{i}{2} K^2, & [X^3, K^3] = \frac{i}{2} K^4, \\
[X^1, K^4] = -\frac{i}{2} K^1, & [X^2, K^4] = -\frac{i}{2} K^1, & [X^3, K^4] = -\frac{i}{2} K^3, \\
[Y^1, K^1] = -\frac{i}{2} K^4, & [Y^2, K^1] = -\frac{i}{2} K^3, & [Y^3, K^1] = \frac{i}{2} K^2, \\
[Y^1, K^2] = \frac{i}{2} K^3, & [Y^2, K^2] = -\frac{i}{2} K^4, & [Y^3, K^2] = -\frac{i}{2} K^1, \\
[Y^1, K^3] = -\frac{i}{2} K^2, & [Y^2, K^3] = \frac{i}{2} K^1, & [Y^3, K^3] = -\frac{i}{2} K^4, \\
[Y^1, K^4] = \frac{i}{2} K^1, & [Y^2, K^4] = \frac{i}{2} K^2, & [Y^3, K^4] = \frac{i}{2} K^3, \\
[K^1, K^2] = i(X^3 + Y^3), & [K^1, K^3] = -i(X^2 + Y^2), & [K^1, K^4] = i(X^1 - Y^1), \\
[K^2, K^3] = i(X^1 + Y^1), & [K^2, K^4] = i(X^2 - Y^2), & [K^3, K^4] = i(X^3 - Y^3).
\end{array}
$$

8.2 Appendix B: Landscape for so(5)

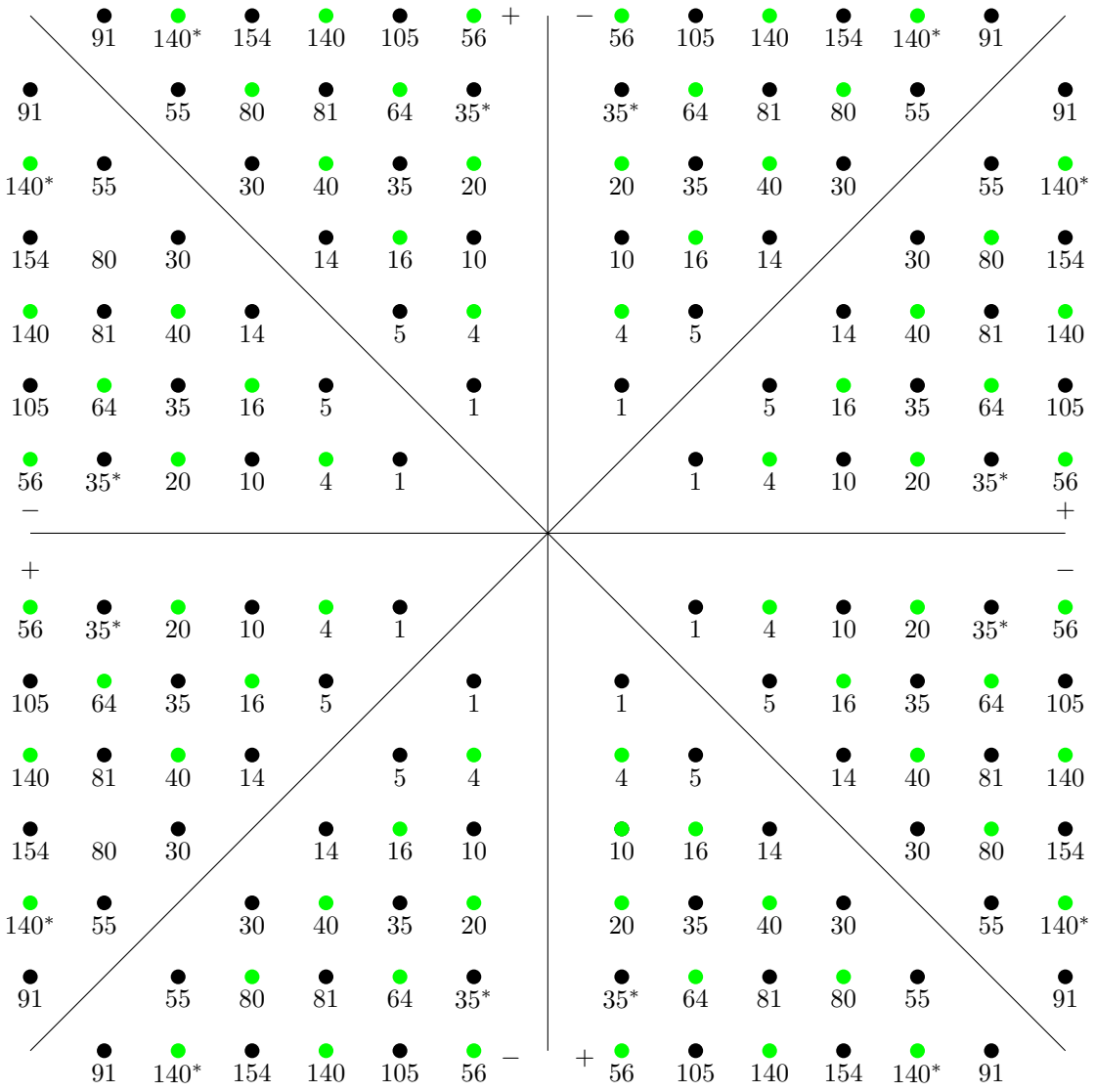


Figure 23: Landscape for $so(5)$.

References

[1] E. Demler, W. Hanke, S.-C. Zhang, *SO(5) Theory of Antiferromagnetism and Superconductivity*, Rev. Mod. Phys. 76 (2004), 909.

[2] A. Bilal, J. P. Derendinger, K. Sfetsos, *(Weak) G_2 Holonomy from Self-duality, Flux and Supersymmetry*, Nucl. Phys. B628 (2002), 112.

[3] J. P. Antoine and D. Speiser, *Characters of Irreducible Representations of the Simple Groups. II. Application to Classical Groups*, J. Math. Phys. 5 (1964), 1560.

[4] J. P. Antoine and D. Speiser, *Characters of Irreducible Representations of the Simple Groups. I. General Theory* J. Math. Phys. 5 (1964), 1226.

[5] M. Böhm, *Lie-Gruppen und Lie-Algebren in der Physik*, Springer Verlag Berlin (2011).

[6] R.E. Behrends, J. Dreitlein, C. Fronsdal and W.Lee, *Simple Groups and Strong Interaction Symmetries*, RevModPhys. 34 (1962), 584.

[7] W. Greiner and B. Müller, *Quantum Mechnics, Symmetries*, Springer Verlag Berlin (1989).

[8] J. Hicklin, C. Moler, P. Webb, JAMA:
`http://math.nist.gov/javanumerics/jama/` [27.06.2016].