REPRESENTATIONS OF LIE ALGEBRAS

WEIZHE SHEN

Abstract. This is an introduction to representation theory of Lie algebras (with a focus on semisimple Lie algebras), together with a short survey on softwares developed for computations in the theory.

Contents

1. Introduction

The theory of Lie algebras and their representations is not only one interesting topic in the mathematical field of representation theory, but also one that has growing applications in various branches of theoretical physics, such as molecular physics and particle physics. Moreover, in modern development of the theory, Lie-algebra related computations and models need to be implemented or built on the computer. Our purpose here is to introduce some major programs and software packages that can be applied to facilitate these computations, as well as the mathematical theory itself.

The paper is organized as follows: Section 2 gives an introduction to the representation theory of Lie algebras, and in particular, the representations of semisimple Lie algebras. (This is a mature theory and usually takes hundreds of pages to cover its foundation. Interested readers are referred to references [3, 4].) Section 3

presents a short survey on four popular software applications that facilitate computations in Lie algebras and their representations. We shall mainly focus on the mathematical parts; features about user experience (namely, how easy and flexible a program is to use), such as supplies of a type checking system, an on-line help system, or a memory management system, are not within the scope of the survey. In Section 4, we discuss the limitations of our work and register our thoughts about some possible future work.

2. Representation Theory of Lie Algebras

In this section we introduce some concepts in the representation theory of Lie algebras, with a focus on finite-dimensional complex semisimple Lie algebras. Unless otherwise specified, the following definitions and results (lemmas, propositions, etc.) are based on the books Fulton-Harris [3] and Humphreys [4].

2.1. **Preliminaries.** A (complex) *Lie algebra* is a (finite-dimensional) complex vector space $\mathfrak{g}$ with a bilinear map

$$[\,\cdot\,,\,\cdot\,] : \mathfrak{g} \times \mathfrak{g} \longrightarrow \mathfrak{g}$$

(usually called *Lie bracket*), satisfying

(1) skew-symmetry: $[x, y] = -[y, x]$, and
(2) Jacobi identity: $[x, [y, z]] + [y, [z, x]] + [z, [x, y]] = 0$,

for all $x, y, z \in \mathfrak{g}$. A subspace $\mathfrak{h}$ of $\mathfrak{g}$ is called a *subalgebra* if $[x, y] \in \mathfrak{h}$ for all $x, y \in \mathfrak{h}$.

A Lie algebra $\mathfrak{g}$ is said to be *nilpotent* if for some $n \in \mathbb{N}$ one has

$$[x_1, [x_2, [\cdots, [x_n, y] \cdots]]] = 0$$

holds for all $x_1, x_2, \cdots, x_n, y \in \mathfrak{g}$.

Example 2.1 (Abelian Lie algebras)**.** If the Lie bracket $[\,\cdot\,,\,\cdot\,]$ of a Lie algebra is trivial (i.e., $[x, y] = 0$ for all x, y), then we say the Lie algebra is *abelian*. It is clear that any abelian Lie algebra is nilpotent.

Example 2.2 ($\mathrm{gl}(V)$)**.** Given a finite-dimensional complex vector space V, $\mathrm{gl}(V)$ is a Lie algebra, where the underlying set is $\mathrm{End}(V)$ (i.e., the set of endomorphisms of V) and the Lie bracket $\mathrm{End}(V) \times \mathrm{End}(V) \to \mathrm{End}(V)$ is defined by $(\alpha, \beta) \mapsto \alpha \circ \beta - \beta \circ \alpha$.

A *Lie algebra homomorphisms* is a linear map $\varphi : \mathfrak{g} \to \mathfrak{h}$ that is compatible with the respective Lie brackets, i.e., for all $x, y \in \mathfrak{g}$ we have

$$\varphi([x, y]_{\mathfrak{g}}) = [\varphi(x), \varphi(y)]_{\mathfrak{h}}.$$

Definition 2.3 (Semisimple Lie algebras)**.** A Lie algebra is said to be *simple* if it contains no nonzero proper ideals. A *semisimple* Lie algebra is a direct sum of simple Lie algebras.

Definition 2.4 (Representations of Lie algebras)**.** A *representation* of a Lie algebra $\mathfrak{g}$ on a complex vector space V is a homomorphism of Lie algebras $\mathfrak{g} \to \mathrm{gl}(V)$.

The following definitions are similar to those in the representation theory of groups: Given a Lie algebra representation $\rho : \mathfrak{g} \to \mathrm{gl}(V)$ and a subspace $W \subset V$, we say (ρ, W) is a *sub-representation* of (ρ, V) if $\rho(x)(W) \subset W$ for all $x \in \mathfrak{g}$. A representation is *irreducible* if it contains no non-trivial sub-representations. A *homomorphism* of Lie algebra representations $\rho_V : \mathfrak{g} \to \mathrm{gl}(V)$ and $\rho_W : \mathfrak{g} \to \mathrm{gl}(W)$ is a linear map $\varphi : V \to W$ such that the following diagram commutes for all $x \in \mathfrak{g}$:

$$\begin{array}{ccc} V & \xrightarrow{\rho_V(x)} & V \\ {\scriptstyle\varphi}\downarrow & & \downarrow{\scriptstyle\varphi} \\ W & \xrightarrow[\rho_W(x)]{} & W \end{array}$$

Example 2.5 (Adjoint representations)**.** The *adjoint representation* of a Lie algebra $\mathfrak{g}$ is a representation of $\mathfrak{g}$ on itself defined by

$$\begin{aligned} \mathrm{ad} : \mathfrak{g} &\longrightarrow \mathrm{gl}(\mathfrak{g}) \\ x &\longmapsto (y \mapsto [x, y]). \end{aligned}$$

Definition 2.6 (Killing forms)**.** Let $\mathfrak{g}$ be a finite-dimensional Lie algebra. For each $x \in \mathfrak{g}$, define $\mathrm{ad}_x : \mathfrak{g} \to \mathfrak{g}$ by

$$\mathrm{ad}_x(y) := [x, y] \quad \text{for all } y \in \mathfrak{g}.$$

The *killing form* on $\mathfrak{g}$ is a symmetric bilinear form κ on $\mathfrak{g}$ defined by

$$\kappa(x, y) := \mathrm{trace}(\mathrm{ad}_x \circ \mathrm{ad}_y).$$

Now we introduce the notion of Weyl groups. Let E be a finite-dimensional Euclidean vector space endowed with a positive definite symmetric bilinear form $(\cdot, \cdot)$. A finite set Φ of E of non-zero vectors is called a *root system* in E if it satisfy the following conditions:

(1) Φ spans E.
(2) For any $\alpha \in \Phi$, the only scalar multiples of α in Φ are $\pm\alpha$.
(3) For any $\alpha, \beta \in \Phi$, the element $\sigma_\alpha(\beta) := \beta - 2\frac{(\beta,\alpha)}{(\alpha,\alpha)}\alpha$ stays in Φ.
(4) For any $\alpha, \beta \in \Phi$, one has $\langle \beta, \alpha \rangle := 2\frac{(\beta,\alpha)}{(\alpha,\alpha)} \in \mathbb{Z}$.

Remark 1. For any $\alpha \in \Phi$, σ_α (as defined in condition (3)) is a reflection through the hyperplane $P_\alpha := \{\beta \in E \mid (\beta, \alpha) = 0\}$.

Definition 2.7 (Weyl groups)**.** A *Weyl group* of a root system Φ in E is the subgroup of $\mathrm{GL}(E)$ generated by $\{\sigma_\alpha \mid \alpha \in \Phi\}$.

Remark 2. Given a semisimple Lie algebra $\mathfrak{g}$ and a Cartan subalgebra $\mathfrak{h}$, the *Weyl group of* $\mathfrak{g}$ is the Weyl group of the root system $(\mathfrak{h}^*)_{\mathrm{ad}}$ in $\mathfrak{h}^*$.

2.2. **An example: $\mathfrak{sl}_2(\mathbb{C})$.** Now we discuss the Lie algebra $\mathfrak{sl}_2(\mathbb{C}) := \{A \in \mathbb{C}^{2\times 2} \mid \mathrm{tr}(A) = 0\}$, which is equipped with Lie bracket $[A, B] := AB - BA$. Consider the following matrices, which form a basis of $\mathfrak{sl}_2(\mathbb{C})$ as a $\mathbb{C}$ vector space:

$$H := \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad X := \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad Y := \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

Given an irreducible representation (ρ, V) of $\mathfrak{sl}_2(\mathbb{C})$, the action of H on V is diagonalizable, and thus V can be decomposed as

$$V = \bigoplus_{\lambda \in \Lambda} V_\lambda,$$

where V_λ's are the eigenspaces. Then for any $\lambda \in \Lambda$ and $v \in V_\lambda$, one has

$$H(X(v)) = [H, X](v) + X(H(v)) = 2X(v) + \lambda X(v) = (\lambda + 2)X(v),$$

and also,

$$H(Y(v)) = [H, Y](v) + Y(H(v)) = -2Y(v) + \lambda Y(v) = (\lambda - 2)Y(v),$$

so $X(v) \in V_{\lambda+2}$ and $Y(v) \in V_{\lambda-2}$. It follows that the direct sum of V_γ's where $\gamma \in \Lambda, \gamma \equiv \lambda \pmod 2$ is a subrepresentation of V. The irreducibility of V implies that $\Lambda = \{\alpha + 2k\}_{k=0}^n$ for some $\alpha \in \Lambda$ and $n \in \mathbb{N}$. Therefore, one has

$$V = \bigoplus_{0 \leqslant k \leqslant n} V_{\alpha+2k}.$$

2.3. **Cartan subalgebras and weight spaces.** In the case of $\mathfrak{sl}_2(\mathbb{C})$, the action of the matrix H on the representations plays a key role in analyzing its irreducible representations. This does not directly apply to general semisimple Lie algebras, but the notion of "maximal" abelian subalgebras, namely *Cartan subalgebras*, can help in a similar way.

Definition 2.8 (Cartan subalgebras)**.** A *Cartan subalgebra* $\mathfrak{h}$ of a Lie algebra $\mathfrak{g}$ is a nilpotent subalgebra satisfying that

$$[x, y] \in \mathfrak{h} \quad \forall x \in \mathfrak{h} \implies y \in \mathfrak{h}.$$

Lemma 2.9. *Every semisimple Lie algebra has a Cartan subalgebra.*

Definition 2.10 (Weights and weight spaces)**.** Let $\mathfrak{g}$ be a semisimple Lie algebra, and let $\mathfrak{h}$ be a Cartan subalgebra of $\mathfrak{g}$. A *weight* of a representation $\rho : \mathfrak{g} \to \mathrm{gl}(V)$ of $\mathfrak{g}$ is a linear functional $\lambda \in \mathfrak{h}^*$ satisfying that

$$\rho(x)(v) = \lambda(x)v, \quad \forall x \in \mathfrak{h}$$

for some $v \in V - \{0\}$. The set of all the possible v's with the zero vector included is called the *weight space* associated to λ and is denoted by V_λ. The *multiplicity* of λ in ρ is defined as the dimension of V_λ.

If $\rho : \mathfrak{g} \to \mathrm{gl}(V)$ is the adjoint representation of $\mathfrak{g}$, then a non-zero weight λ of ρ is called a *root* of the Lie algebra $\mathfrak{g}$, and the associated weight space is called the *root space* of λ.

Theorem 2.11 (Cartan decomposition)**.** *Let $\mathfrak{g}$ be a semisimple Lie algebra, and let $\mathfrak{h}$ be a Cartan subalgebra of $\mathfrak{g}$. Then $\mathfrak{g}$ can be decomposed as*

$$\mathfrak{g} = \mathfrak{h} \oplus \bigoplus_{\lambda \in (\mathfrak{h}^*)_{ad}} \mathfrak{g}_\lambda,$$

where $(\mathfrak{h}^*)_{ad}$ denotes the set of roots of $\mathfrak{g}$ corresponding to $\mathfrak{h}$, and $\mathfrak{g}_\lambda$ denotes the root space of λ.

Theorem 2.12 (Weight space decomposition)**.** *Let $\mathfrak{g}$ be a semisimple Lie algebra, and let $\mathfrak{h}$ be a Cartan subalgebra of $\mathfrak{g}$. Given a representation $\rho : \mathfrak{g} \to gl(V)$, V can be decomposed as*

$$V = \bigoplus_{\lambda \in (\mathfrak{h}^*)_\rho} V_\lambda,$$

where $(\mathfrak{h}^*)_\rho$ denotes the set of the weights of ρ corresponding to $\mathfrak{h}$.

2.4. **A procedure for describing representations.** Here we describe the outline of a procedure for classifying irreducible, finite-dimensional representations of semisimple Lie algebras. Some relevant facts together with a more detailed algorithmic procedure can be found in Lecture 14 of [3].

First of all, we need to verify that the Lie algebra being analyzed, say $\mathfrak{g}$, is semisimple. Let V be any representation of $\mathfrak{g}$. The subsequent steps are listed as follows:

(1) Analogous to finding the matrix H in the case of $\mathfrak{sl}_2(\mathbb{C})$, find an abelian subalgebra $\mathfrak{h}$ of $\mathfrak{g}$ whose action on the representations of $\mathfrak{g}$ is diagonal. (In fact, it suffices to look for an abelian subalgebra that acts diagonally on one faithful representation; see Theorem 9.20 in [3].) Since a larger abelian subalgebra would provide more information on the structure of representations, we consider the "maximal" case, namely a Cartan subalgebra.
(2) Decompose $\mathfrak{g}$ with respect to the abelian, diagonalizable subalgebra $\mathfrak{h}$ found in step (1).
(3) Find the copies of subalgebras $\mathfrak{sl}_2(\mathbb{C})$ (i.e., the subalgebras which are isomorphic to $\mathfrak{sl}_2(\mathbb{C})$) contained in $\mathfrak{g}$.
(4) For each copy, pick a basis H, X, Y.
(5) Apply the integrality of the eigenvalues of the H's and form the weight lattice of $\mathfrak{g}$.
(6) Apply the semmetry of the eigenvalues of the H's and form the Weyl group of $\mathfrak{g}$. Break up the eigenspace decomposition of V.
(7) Introduce the killing form on $\mathfrak{g}$.
(8) Choose a real linear functional in $\mathfrak{h}^*$ that is irrational on the weight lattice.
(9) Analyze.

A concrete example applying the above procedure to the representations of $\mathfrak{sl}_3(\mathbb{C})$ can be found in Lectures 12-13 of [3].

Remark 3. If the Lie algebra being analyzed is not semisimple, the above procedure still give information about the irreducible representations: we can use it to the semisimple part of the given Lie algebra. The theoretical support can be found in Lecture 9 of [3].

3. Programs and Software Packages

In this section we focus on analyzing four popular programs (or software packages): LiE [8], SimpLie [6], Affine.m [5], and LieART [1]. We will also discuss their limitations (if any) and in what manners problems can be solved (if the computational algorithm is inspiring or interesting). The reader who is interested in

designing similar programs or needs to frequently apply functions for computations related to these topics may also find these two packages helpful: [2] and [7].

The programs are introduced in chronological order (according to the time they were published).

3.1. **Basic computational questions.** First, we discuss some basic computational questions encountered in Lie algebras and their representations that can be handled computationally. To begin with, some mathematical objects, such as the decomposition polynomial of an adjoint representation, the dimension of a representation, and the product of irreducible representations, can be directly returned with appropriate inputs into some built-in functions. More importantly, the main computational questions (as mentioned in [5]) are

(1) Construction of a root system which determines the properties of Lie algebra including its commutation relations.
(2) Weyl group traversal which is important due to Weyl symmetry of root system and characters of representations.
(3) Calculation of weight multiplicities, branching and fusion coefficients, which are essential for construction and study of representations.

3.2. **LiE.** [8, 9] LiE is a computer algebra package for computations with Lie groups, Lie algebras, and their representations. There are about one-hundred mathematical functions built into the library of LiE. These functions implement operations related to Lie groups and algebras, the structure of their Weyl groups and root systems, symmetric groups, and representations. Most of the built-in functions take a group as parameter. Standard classification of simple Lie groups is pre-defined in the package, and accordingly, semisimple groups are formed by concatenation and reductive groups are formed by adding a complex torus.

It is important to note that LiE is mainly designed for semisimple Lie groups and algebras. It works with reductive groups, and only simply connected simple groups are given names. This in general does not cause a limitation. Readers who are concerned about the class of groups and need more information to compare with their actual needs can see pages 3 and 37 in [9].

One interesting feature of LiE is that it has its own programming language, so in addition to calling the built-in operations and functions, users can define new algorithms. In other words, the library of functions is not fixed or exclusive to its developers; it can be extended or customized by its users.

LiE can also do ordinary arithmetic and accepts large magnitude of numbers (so there is no bound in practical applications), but only with integral numbers, matrices (or vectors) with integral entries, and polynomials with integral coefficients. LiE is not a symbolic package, in the sense that formal symbols are not defined. Although the data types is few, they are stored efficiently.

LiE is written in C and is available on many platforms (e.g., systems running UNIX or with a C-compiler).

3.3. **SimpLie.** [6] There are two specialties of SimpLie: one is that it handles problems arising in infinite-dimensional Lie algebras (later we will see that Affine.m is designed for finite-dimensional and affine Lie algebras), and secondly, it "visualizes" Lie algebras by presenting screenshots of Coxeter plane projection or Hasse diagrams of the root system of the algebra.

Other major features of SimpLie consist of calculation of Lie algebra properties, such as dimension, rank, and roots (based on Dynkin diagrams), calculation of highest weight representations, and level decompositions of Lie algebras.

SimpLie is written in Java, and is also available on different types of computers (or platforms).

3.4. **Affine.m.** [5] Affine.m is a MATHEMATICA package. As we mentioned above, it is developed for handling problems frequently encountered in representation theory of affine and finite-dimensional Lie algebras. In particular, the developer aimed to extend the tables of multiplicities of affine Lie algebras and representations. Furthermore, Affine.m has a focus on physical applications, and the problems it deals with include but not limited to the following [5]:

(1) tensor product decomposition for finite-dimensional Lie algebras,
(2) branching and parabolic Verma modules,
(3) string functions of affine Lie algebras and CFT models, and
(4) branching functions and coset models of conformal field theory.

Now we elaborate a little on the computational algorithms of Affine.m. To begin with, its algorithms for finite-dimensional Lie algebras were adapted from the structure of affine Lie algebras. For some computational problems, e.g., construction of a root system, Affine.m was based on existing well-known algorithms. For the most computationally intensive problems–calculation of weight multiplicities, branching, and fusion coefficients–Affine.m analyzed two recurrent relations/algorithms. The reader who is interested in developing software with the similar aim can see pages 17-20 in [5] for more details.

3.5. **LieART.** [1] LieART is the most recently published program that we discuss in this paper. LieART basically covers all the Lie-algebra related computations that can be achieved by the previous programs, for example, tensor product decomposition, computations of root systems of Lie algebras, generation of weight system of an irreducible representation, etc.

Similar to Affine.m, one special property of LieART is that, it has a flavor of physics, especially particle physics, so physicists who require the use of Lie algebras and their representations may find these two programs more helpful.

The development of LieART was intended to bring Lie-algebra related software to the computer-algebra system MATHEMATICA. Moreover, it enriched the tables of irreducible representations and their invariants.

LieART is a MATHEMATICA application, and it is also cross-platform.

4. Conclusion

In conclusion, we introduced some basic concepts and notions in Lie algebras and their representation and analyzed four popular softwares designed for computations arising in the theory.

There are some limitations of our work. To begin with, the overview of the theoretical part has been significantly narrowed, and there are a number of results that we did not get to present/prove in this paper. Also, due to the lack of a computer at hand during the remote-instruction period, the author was not able to test any of the programs or software packages discussed above; all of the analyses presented in Section 3 were based on the examples provided by the developers of the corresponding softwares.

Future work could further investigate these softwares so that a more complete survey could be presented. These softwares could also be explored in a way that Lie algebra representations could be viewed from a programming perspective and thus to see if some existing theoretical ideas could be improved.

While brief, we hope this small piece of writing encourages the readers to further explore the applications of Lie algebra representations.

References

[1] Feger, R., & Kephart, T. W. (2015). LieART–a Mathematica application for Lie algebras and representation theory. *Computer Physics Communications*, 192, 166-195. URL: https://arxiv.org/abs/1206.6379.

[2] Fischbacher, T. (2002). Introducing LambdaTensor1.0 - A package for explicit symbolic and numeric Lie algebra and Lie group calculations. URL: https://arxiv.org/abs/hep-th/0208218.

[3] Fulton, W., & Harris, J. (2013). *Representation theory: a first course* (Vol. 129). Springer Science & Business Media.

[4] Humphreys, J. E. (2012). *Introduction to Lie algebras and representation theory* (Vol. 9). Springer Science & Business Media.

[5] Nazarov, A. (2012). Affine.m–Mathematica package for computations in representation theory of finite-dimensional and affine Lie algebras. *Computer Physics Communications*, 183(11), 2480-2493. URL: https://arxiv.org/abs/1107.4681.

[6] Nutma, T. (2009). SimpLie. URL: http://code.google.com/p/simplie/.

[7] Stembridge, J. R. (1995). A Maple package for symmetric functions. *Journal of Symbolic Computation*, 20(5-6), 755-758.

[8] Van Leeuwen, M. A., Cohen, A. M., & Lisser, B. (1992). LiE: A package for Lie group computations. Computer Algebra, Nederland, Amsterdam. URL: http://young.sp2mi.univ-poitiers.fr/~marc/LiE/.

[9] Van Leeuwen, M. A., Cohen, A. M., & Lisser, B. (1992). LiE Manual. URL: http://www-math.univ-poitiers.fr/~maavl/LiEman/manual.pdf.