FinLie

Computation with finite Lie algebras

A GAP 4 package

by

Bettina Eick
Institut Computational Mathematics
TU Braunschweig

Contents

1 Introduction

This package contains methods to compute with finite Lie algebras; that is, with Lie algebras over finite fields. The functions in this package are often particularly effective if the considered Lie algebras are solvable, since there are various special methods for finite solvable Lie algebras implemented.

The GAP system contains various methods to compute with Lie algebras, see [DGr00] for background. This package is based on these methods and extends them in the case of a finite Lie algebra, see [Eic04a] for a description of some of the methods.

2 Solvable Lie algebras

First, we describe some of the basic functions for finite solvable Lie algebras which are used throughout the package.

1 ► `LowerNilpotentBasis(`L`)`

Takes as input a solvable Lie algebra L and returns a record with entries *basis* and *weights*. The entry *basis* is a list of elements in L which forms a basis. The basis exhibits the lower nilpotent series of L which is a automorphism-invariant series with abelian factors. The entry *weights* is a list which describes the steps of the series.

2 ► `UpperNilpotentBasis(`L`)`

Takes as input a solvable Lie algebra L and returns a record with entries *basis* and *weights*. The entry *basis* is a list of elements in L which forms a basis. The basis exhibits the upper nilpotent series of L which is a automorphism-invariant series with abelian factors. The entry *weights* is a list which describes the steps of the series.

3 Cohomology and extensions for finite Lie algebras

1 ► `LieOneCobounds(` L, B, M `)`
► `LieOneCocycles(` L, B, M `)`
► `LieOneCohomology(` L, B, M `)`

These functions take as input a Lie algebra L, a basis B of L and a list of matrices M such that the map mapping the elements of B to the elements of M induces a representation of L.

Let $B = \{b_1, \ldots, b_n\}$ and let V be the natural L-module defined by the given representation. Then the map $Z^1(L, V) \to V^n : \varphi \mapsto (b_1^\varphi, \ldots, b_n^\varphi)$ is a faithful representation of $Z^1(L, V)$.

The functions LieOneCobounds and LieOneCocyles return lists of vectors which form a basis for the images of the representations of $Z^1(L, V)$ and $B^1(L, V)$, respectively.

The function LieOneCohomology returns a linear map onto the factor of the results of LieOneCocycles modulo LieOneCobounds. Thus the image of this map is a faithful representation of $H^1(L, V)$.

2 ► `LieTwoCobounds(` L, B, M `)`
► `LieTwoCocycles(` L, B, M `)`
► `LieTwoCohomology(` L, B, M `)`

These functions take as input a Lie algebra L, a basis B of L and a list of matrices M such that the map mapping the elements of B to the elements of M induces a representation of L.

The functions LieTwoCobounds and LieTwoCocyles return lists of vectors which form a basis for the images of representations of $Z^2(L, V)$ and $B^2(L, V)$, respectively. The chosen representations may not be faithful!

The function LieTwoCohomology returns a linear map onto the factor of the results of LieTwoCocycles modulo LieTwoCobounds. The image of this map is a faithful representation of $H^2(L, V)$.

3 ► `LieExtensionByCocycle(` L, B, M, c `)`

This function takes as input a Lie algebra L, a basis B of L, a list of matrices M such that the map mapping the elements of B to the elements of M induces a representation of L and a 2-cocyle c in the representation ψ of $Z^2(L, V)$.

The function returns a Lie algebra which is an extension of L by the natural L-module V defined by the representation via c.

4 Automorphism groups and isomorphism testing

1 ▶ `AutomorphismGroupOfSolvableLieAlgebraBySpecial(`L`)`
▶ `AutomorphismGroupOfSolvableLieAlgebraByFitting(`L`)`
▶ `AutomorphismGroupOfSolvableLieAlgebra(`L`)`
▶ `AutomorphismGroupOfLieAlgebra(`L`)`

All of these functions take as input a finite Lie algebra L. In the first three cases L must be solvable. All functions return the automorphism group of L. The last function can be really slow if L is not solvable.

2 ▶ `AutomorphismGroup(`L`)` A

Computes the automorphism group of the finite Lie algebra L and stores it as an attribute.

3 ▶ `CanonicalFormOfSolvableLieAlgebra(`L`)` A

Takes as input a finite solvable Lie algebra L and it returns another finite solvable Lie algebra K isomorphism to L whose structure constants are in a canonical form. Two Lie algebras L and H are isomorphic if and only if the structure constants of their canonical forms are equal. The computed canonical form is stored as an attribute.

4 ▶ `AreIsomorphicLieAlgebras(`L`, `H`)` M

This function checks whether the finite Lie algebras L and H are isomorphic. The function can be really slow if L or H are not solvable.

Bibliography