

# **LAG**

---

## **A GAP4 Package**

**For Lie Algebras of Group Algebras  
and other Associative Algebras**

**Version 2.1**

**by**

**Richard Rossmanith**

Friedrich-Schiller-Universität Jena  
D-07740 Jena, Germany

**April 2002**

# Contents

<b>1</b>	<b>The LAG Package</b>	<b>3</b>
1.1	Attributes, Properties and Operations . . . . .	3
	<b>Bibliography</b>	<b>11</b>
	<b>Index</b>	<b>12</b>

# 1

# The LAG Package

The LAG package contains the declaration of attributes, properties, and operations for associated Lie algebras of associative algebras, in particular group algebras.

If  $A$  is an associative algebra, its associated Lie algebra is the Lie Algebra  $L$  which has the same underlying vector space as  $A$ , and which satisfies  $[a, b] = ab - ba$  for all  $a, b \in A$ .

In GAP, however, the bracket notation  $[a, b]$  is reserved for lists, so the product in  $L$  is denoted by the star  $*$  – the same symbol which is also used to denote the associative multiplication in  $A$ . Therefore, GAP needs to distinguish between the elements in  $A$  and the elements in  $L$ , i.e. here the underlying vector spaces are not equal, but only (however canonically) isomorphic. GAP stores the Lie algebra  $L$  and the natural linear bijection from  $A$  onto  $L$  as attributes of the associative algebra  $A$ .

The usual commands that apply to algebras, such as `Dimension`, `IsFiniteDimensional`, `IsFinite`, `Size`, `Elements`, etc. also work for the Lie Algebra, if they work for the underlying associative algebra. Additionally, the standard Lie algebra functions (described in the chapter about Lie algebras) of course also apply to Lie algebras that come from associative algebras. This will not be explained in detail for every single command in this chapter.

The main objective of this package, however, is to deal with Lie algebras of group algebras. Some new functions are added, and, for other functions that also apply to abstract Lie algebras, much faster methods are implemented (which was possible due to the special structure of such Lie algebras).

I would like to point out that many properties of Lie algebras of group algebras carry over to the commutator structure of the unit group. E.g., if the Lie algebra of the group algebra is solvable, then so is its unit group (in odd characteristic — there are counterexamples in characteristic 2). However, no such functions have been included in the LAG package, due to the fact that unit groups were only “in the making” at the time when LAG was programmed. See the survey article [Bov98] for a detailed description of the interplay between the unit group and the Lie algebra of a group algebra. It might be useful for future implementations of fast algorithms for the unit group.

The LAG package arose as a byproduct of the author’s dissertation thesis [Ros97]. It was ported to GAP 4 and brought into standard GAP package format during a visit to St. Andrews in September 1998, under the supervision of the GAP team. I want to thank everybody on the team for their support, in particular Steve Linton, Willem de Graaf, Thomas Breuer, and Alexander Hulpke.

(Richard Rossmanith)

## 1.1 Attributes, Properties and Operations

1 ► `LieAlgebraByDomain( A )`

M

This method takes an associative algebra as argument, and constructs its associated Lie algebra. The user, however, will **never use** this command, but will rather use `LieAlgebra( A )`, which either returns the Lie algebra in case it is already constructed, or refers to `LieAlgebraByDomain` in case it is not.

```
gap> M:=MatrixAlgebra(GF(3),3);
( GF(3)^[ 3, 3 ] )
gap> L:=LieAlgebra(M);
<Lie algebra over GF(3)>
```

## 2 ► IsLieAlgebraByAssociativeAlgebra( L )

C

This category signifies that the Lie algebra is constructed as an associated Lie algebra of an associative algebra. (That knowledge cannot be obtained later on.)

```
gap> M:=MatrixAlgebra(GF(3),3);
( GF(3)^[ 3, 3 ] )
gap> L:=LieAlgebra(M);
<Lie algebra over GF(3)>
gap> IsLieAlgebraByAssociativeAlgebra(L);
true
```

## 3 ► UnderlyingAssociativeAlgebra( L )

A

If a Lie algebra is constructed from an associative algebra, it remembers this underlying associative algebra as one of its attributes.

```
gap> M:=MatrixAlgebra(GF(3),3);
<algebra-with-one over GF(3), with 2 generators>
gap> L:=LieAlgebra(M);
<Lie algebra over GF(3)>
gap> UnderlyingAssociativeAlgebra(L);
( GF(3)^[ 3, 3 ] )
gap> last=M;
true
```

## 4 ► NaturalBijectionToLieAlgebra( A )

A

The natural linear bijection between the (isomorphic, but not equal, see introduction) underlying vector spaces of an associative algebra  $A$  and its associated Lie algebra is stored as an attribute of  $A$ . (Note that this is a vector space isomorphism between two algebras, but not an algebra homomorphism in general.)

```
gap> F:=GF(2); G:=SymmetricGroup(3); FG:=GroupRing(F,G);
GF(2)
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
gap> t:=NaturalBijectionToLieAlgebra(FG);
MappingByFunction( <algebra-with-one over GF(2), with
2 generators>, <Lie algebra over GF(
2)>, <Operation "LieObject">, function( y ) ... end )
gap> a:=Random(FG);
(Z(2)^0)*(1,2,3)+(Z(2)^0)*(1,3,2)+(Z(2)^0)*(1,3)
gap> a*a;
# product in the associative algebra
(Z(2)^0)*()+(Z(2)^0)*(1,2,3)+(Z(2)^0)*(1,3,2)
gap> b:=a^t;
LieObject( (Z(2)^0)*(1,2,3)+(Z(2)^0)*(1,3,2)+(Z(2)^0)*(1,3) )
gap> b*b;
# product in the Lie algebra (commutator) ...
LieObject( <zero> of ... )
# ... must be zero!
```

## 5 ► NaturalBijectionToAssociativeAlgebra( L )

A

This is the inverse of the linear bijection mentioned above, stored as an attribute of the Lie algebra.

```

gap> G:=SymmetricGroup(3); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
<Lie algebra over GF(2)>
gap> s:=NaturalBijectionToAssociativeAlgebra(L);
MappingByFunction( <Lie algebra over GF(2)>, <algebra-with-one over GF(
2), with 2 generators>, function( y ) ... end, <Operation "LieObject"> )
gap> InverseGeneralMapping(s)=NaturalBijectionToLieAlgebra(FG);
true

```

#### 6 ► IsLieAlgebraOfGroupRing( $L$ )

P

If a Lie algebra is constructed from an associative algebra which happens to be in fact a group ring, it has many nice properties that can be used for faster algorithms, so this information is stored as a property.

```

gap> F:=GF(2); G:=SymmetricGroup(3); FG:=GroupRing(F,G);
GF(2)
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
gap> L:=LieAlgebra(FG);
<Lie algebra over GF(2)>
gap> IsLieAlgebraOfGroupRing(L);
true

```

#### 7 ► UnderlyingGroup( $L$ )

A

The underlying group of a Lie algebra  $L$  which is constructed from a group ring is defined to be the underlying magma of the group ring in question. (Note that the term **underlying group** applies only to the Lie algebra. For the group ring itself, the term **underlying magma** is preferred over **underlying group** for greater generality. In fact (or unfortunately?), “underlying group of a group ring” is not at all defined.)

**Remark:** The underlying field may be accessed by the command `LeftActingDomain(  $L$  )`.

```

gap> F:=GF(2); G:=SymmetricGroup(3); FG:=GroupRing(F,G);
GF(2)
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
gap> L:=LieAlgebra(FG);
<Lie algebra over GF(2)>
gap> UnderlyingGroup(L);
Sym( [ 1 .. 3 ] )
gap> LeftActingDomain(L);
GF(2)

```

#### 8 ► NaturalMapping( $U$ , $L$ )

O

Let  $U$  be a submagma of a group  $G$ , let  $A := FG$  be the group ring of  $G$  over some field  $F$ , and let  $L$  be the associated Lie algebra of  $A$ . Then `NaturalMapping(  $U$ ,  $L$  )` returns the obvious mapping  $U \rightarrow L$  (as the composition of the mappings `Embedding(  $U$ ,  $A$  )` and `NaturalBijectionToLieAlgebra(  $A$  )`).

```

gap> F:=GF(2); G:=SymmetricGroup(3); FG:=GroupRing(F,G);
GF(2)
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
gap> L:=LieAlgebra(FG);
<Lie algebra over GF(2)>

```

```

gap> f:=NaturalMapping(G,L);
CompositionMapping( MappingByFunction( <algebra-with-one over GF(2), with
2 generators>, <Lie algebra over GF(
2)>, <Operation "LieObject">, function( y ) ... end ), <mapping: SymmetricGrou\
p( [ 1 .. 3 ] ) -> AlgebraWithOne( GF(2), ... ) > )
gap> (1,2)^f + (1,3)^f;
LieObject( (Z(2)^0)*(1,2)+(Z(2)^0)*(1,3) )

```

#### 9 ► AugmentationHomomorphism( A )

A

Nomen est omen for this attribute of the group ring  $A := FG$  of a group  $G$  over some field  $F$ .

```

gap> F:=GF(2); G:=SymmetricGroup(3); FG:=GroupRing(F,G);
GF(2)
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
gap> e:=Embedding(G,FG);
<mapping: SymmetricGroup( [ 1 .. 3 ] ) -> AlgebraWithOne( GF(2), ... ) >
gap> x:=(1,2)^e; y:=(1,3)^e;
Z(2)^0*(1,2)
Z(2)^0*(1,3)
gap> a:=AugmentationHomomorphism(FG);
[ Z(2)^0*(1,2,3), Z(2)^0*(1,2) ] -> [ Z(2)^0, Z(2)^0 ]
gap> x^a; y^a; (x+y)^a;
Z(2)^0
Z(2)^0
0*Z(2)

```

#### 10 ► LieCentre( L )

M

The (Lie) centre of a Lie algebra of a group ring corresponds to the centre of the underlying group ring, and it can be calculated very fast by considering the conjugacy classes of the group. Since the corresponding method for the centre of the group ring does just that, it is being referred to by the method at hand.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning. This is particularly important for the command `LieCentre`.

```

gap> G:=SmallGroup(256,400); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
<pc group of size 256 with 8 generators>
<algebra-with-one over GF(2), with 8 generators>
<Lie algebra over GF(2)>
gap> C:=LieCentre(L);
<Lie algebra of dimension 28 over GF(2)>
gap> D:=LieDerivedSubalgebra(L);
<Lie algebra of dimension 228 over GF(2)>
gap> c:=Dimension(C); d:=Dimension(D); l:=Dimension(L);
28
228
256
gap> c+d=l;
true      # This is always the case for Lie algebras of group algebras!

```

#### 11 ► LieDerivedSubalgebra( L )

M

The (Lie) derived subalgebra of a Lie algebra of a group ring can be calculated very fast by considering the conjugacy classes of the group.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning.

```
gap> G:=SmallGroup(256,400); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
<pc group of size 256 with 8 generators>
<algebra-with-one over GF(2), with 8 generators>
<Lie algebra over GF(2)>
gap> C:=LieCentre(L);
<Lie algebra of dimension 28 over GF(2)>
gap> D:=LieDerivedSubalgebra(L);
<Lie algebra of dimension 228 over GF(2)>
gap> l:=Dimension(L); c:=Dimension(C); d:=Dimension(D);
256
28
228
gap> c+d=l;
true          # This is always the case for Lie algebras of group algebras!
```

12 ► `IsLieAbelian( L )`

M

The Lie algebra  $L$  of an associative algebra  $A$  is Lie abelian, if and only if  $A$  is abelian, so this method refers to `IsAbelian( A )`.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning. This is particularly important for the command `IsLieAbelian`.

```
gap> G:=SymmetricGroup(3); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
<Lie algebra over GF(2)>
gap> IsAbelian(G);
false
gap> IsAbelian(L);      # This command should never be used for Lie algebras!
true                   # It gives a result, but (probably) not the desired one.
gap> IsLieAbelian(L);   # Instead, IsLieAbelian is the correct command.
false
```

13 ► `IsLieSolvable( L )`

M

Passi, Passman and Sehgal [PPS73] have classified all groups  $G$  such that the associated Lie algebra  $L$  of the group ring is (Lie) solvable. This method uses their classification, making it considerably faster than the more elementary method which just calculates Lie commutators.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning.

```
gap> G:=SmallGroup(256,400); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
<pc group of size 256 with 8 generators>
<algebra-with-one over GF(2), with 8 generators>
<Lie algebra over GF(2)>
gap> IsLieSolvable(L);          # This is very fast.
true
gap> List(LieDerivedSeries(L), Dimension); # This is very slow.
```

[ 256, 228, 189, 71, 0 ]

14 ► `IsLieNilpotent( L )`

M

Passi, Passman and Sehgal [PPS73] have classified all groups  $G$  such that the associated Lie algebra  $L$  of the group ring is (Lie) nilpotent. This method uses their classification, making it considerably faster than the more elementary method which just calculates Lie commutators.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning.

```
gap> G:=SmallGroup(256,400); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
<pc group of size 256 with 8 generators>
<algebra-with-one over GF(2), with 8 generators>
<Lie algebra over GF(2)>
gap> IsLieNilpotent(L);
true
# This is very fast.
gap> List(LieLowerCentralSeries(L), Dimension); # This is very slow.
[ 256, 228, 222, 210, 191, 167, 138, 107, 76, 54, 29, 15, 6, 0 ]
```

15 ► `IsLieMetabelian( L )`

P

A Lie algebra is called (Lie) metabelian, if its (Lie) derived subalgebra is (Lie) abelian, i.e. its second (Lie) derived subalgebra is trivial.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning.

Levin and Rosenberger [LR86] have classified all groups  $G$  such that the associated Lie algebra  $L$  of the group ring is (Lie) metabelian. This method uses their classification, making it considerably faster than the more elementary method which just calculates Lie commutators.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning.

```
gap> G:=SmallGroup(256,400); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
<pc group of size 256 with 8 generators>
<algebra-with-one over GF(2), with 8 generators>
<Lie algebra over GF(2)>
gap> IsLieMetabelian(L);
false
```

16 ► `IsLieCentreByMetabelian( L )`

P

A Lie algebra is called (Lie) centre-by-metabelian, if its second (Lie) derived subalgebra is contained in the (Lie) centre of the Lie algebra.

Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning.

In various papers, Külshammer, Sahai, Sharma, Srivastava and the author of the LAG package have classified all groups  $G$  such that the associated Lie algebra  $L$  of the group ring is (Lie) centre-by-metabelian. The most general result to date may be found in [Ros00]. This method uses the classification, making it considerably faster than the more elementary method which just calculates Lie commutators.



Note that the prefix `Lie` is consistently used to distinguish properties of Lie algebras from the analogous properties of groups (or of general algebras). Not using this prefix may result in error messages, or even in wrong results without warning.

```
gap> G:=SymmetricGroup(3); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
<Lie algebra over GF(2)>
gap> IsLieMetabelian(L);
false
gap> IsLieCentreByMetabelian(L);
true
```

#### 17 ► SubgroupsOfIndexTwo( *G* )

A

A list is returned here. (The subgroups of index two in the group  $G$  are important for the Lie structure of the group algebra  $FG$ , in case that the underlying field  $F$  has characteristic 2.)

#### 18 ► CanonicalBasis( *L* )

M

This method directly computes the canonical basis of the Lie algebra of a group algebra without referring to the group algebra, i.e. by sending the group elements directly to the Lie algebra.

```
gap> G:=SymmetricGroup(3); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
<Lie algebra over GF(2)>
gap> L:=LieAlgebra(FG);
<Lie algebra over GF(2)>
gap> B:=CanonicalBasis(L);
CanonicalBasis( <Lie algebra of dimension 6 over GF(2)> )
gap> Elements(B);
[ LieObject( Z(2)^0*() ), LieObject( Z(2)^0*(2,3) ),
  LieObject( Z(2)^0*(1,2) ), LieObject( Z(2)^0*(1,2,3) ),
  LieObject( Z(2)^0*(1,3,2) ), LieObject( Z(2)^0*(1,3) ) ]
```

#### 19 ► IsBasisOfLieAlgebraOfGroupRing( *B* )

P

A basis has this property if the basis vectors are exactly the images of group elements (in sorted order). A basis can be told that it has this above property. (This is important for the speed of the calculation of the structure constants table.)

```
gap> G:=SymmetricGroup(3); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
Sym( [ 1 .. 3 ] )
<algebra-with-one over GF(2), with 2 generators>
<Lie algebra over GF(2)>
gap> L:=LieAlgebra(FG);
<Lie algebra over GF(2)>
gap> B:=CanonicalBasis(L);
CanonicalBasis( <Lie algebra of dimension 6 over GF(2)> )
gap> IsBasisOfLieAlgebraOfGroupRing(B);
true
```

#### 20 ► StructureConstantsTable( *B* )

M

A very fast implementation for calculating the structure constants of a Lie algebra of a group ring w.r.t. its canonical basis  $B$  by using the special structure of  $B$ .

```

gap> G:=CyclicGroup(2); FG:=GroupRing(GF(2),G); L:=LieAlgebra(FG);
<pc group of size 2 with 1 generators>
<algebra-with-one over GF(2), with 1 generators>
<Lie algebra over GF(2)>
gap> L:=LieAlgebra(FG);
<Lie algebra over GF(2)>
gap> B:=CanonicalBasis(L);
CanonicalBasis( <Lie algebra of dimension 2 over GF(2)> )
gap> StructureConstantsTable(B);
[ [ [ [ ], [ ] ], [ [ ], [ ] ] ], [ [ [ ], [ ] ], [ [ ], [ ] ] ], -1,
  0*Z(2) ]

```

# Bibliography

- [Bov98] Adalbert Bovdi. The group of units of a group algebra of characteristic  $p$ . *Publ. Math. Debrecen*, 52(1-2):193–244, 1998.
- [LR86] F. Levin and G. Rosenberger. Lie metabelian group rings. In *Group and semigroup rings (Johannesburg, 1985)*, pages 153–161. North-Holland, Amsterdam, 1986.
- [PPS73] I. B. S. Passi, D. S. Passman, and S. K. Sehgal. Lie solvable group rings. *Canad. J. Math.*, 25:748–757, 1973.
- [Ros97] Richard Rossmanith. *Centre-by-metabelian group algebras*. PhD thesis, Friedrich-Schiller-Universität Jena, 1997.
- [Ros00] Richard Rossmanith. Lie centre-by-metabelian group algebras in even characteristic. I, II. *Israel J. Math.*, 115:51–75, 77–99, 2000.

# Index

This index covers only this manual. A page number in *italics* refers to a whole section which is devoted to the indexed subject. Keywords are sorted with case and spaces ignored, e.g., “PermutationCharacter” comes before “permutation group”.

## A

Attributes, Properties and Operations, *3*  
AugmentationHomomorphism, 6

## C

CanonicalBasis, 9

## I

IsBasisOfLieAlgebraOfGroupRing, 9  
IsLieAbelian, 7  
IsLieAlgebraByAssociativeAlgebra, 3  
IsLieAlgebraOfGroupRing, 5  
IsLieCentreByMetabelian, 8  
IsLieMetabelian, 8  
IsLieNilpotent, 7  
IsLieSolvable, 7

## L

LieAlgebraByDomain, 3  
LieCentre, 6  
LieDerivedSubalgebra, 6

## N

NaturalBijectionToAssociativeAlgebra, 4  
NaturalBijectionToLieAlgebra, 4  
NaturalMapping, 5

## S

StructureConstantsTable, 9  
SubgroupsOfIndexTwo, 9

## U

UnderlyingAssociativeAlgebra, 4  
UnderlyingGroup, 5