

# **LiePRing**

# **A GAP4 Package**

**Version 2.9.1**

**by**

**Bettina Eick (Braunschweig)**  
**and**  
**Michael Vaughan-Lee (Oxford)**

# Contents

<b>1</b>	<b>Preamble</b>	<b>3</b>
<b>2</b>	<b>Lie p-rings</b>	<b>4</b>
<b>3</b>	<b>LiePRings in GAP</b>	<b>5</b>
3.1	Ordinary Lie p-rings . . . . .	5
3.2	Generic Lie p-rings . . . . .	6
3.3	Specialising Lie $p$ -rings . . . . .	7
3.4	Subrings of Lie p-rings . . . . .	9
3.5	Elementary functions . . . . .	10
3.6	Series of subrings . . . . .	10
3.7	The Lazard correspondence . . . . .	11
<b>4</b>	<b>The Database</b>	<b>12</b>
4.1	Accessing Lie p-rings . . . . .	12
4.2	Numbers of Lie p-rings . . . . .	14
4.3	Searching the database . . . . .	14
4.4	More details . . . . .	15
4.5	Special functions for dimension 7 . . . . .	16
4.6	Dimension 8 and maximal class . . . . .	17
<b>5</b>	<b>Advanced functions for Lie p-rings</b>	<b>18</b>
5.1	Schur multipliers . . . . .	18
5.2	Automorphism groups . . . . .	19
	<b>Bibliography</b>	<b>21</b>
	<b>Index</b>	<b>22</b>

# 1

# Preamble

**Abstract:** This package gives access to the database of Lie  $p$ -rings of order at most  $p^7$  as determined by Mike Newman, Eamonn O'Brien and Michael Vaughan-Lee, see [NOVL03] and [OVL05], and it provides some functionality to work with these Lie  $p$ -rings.

**Copyright:** The LieP Ring package is free software; you can redistribute it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your opinion) any later version. The LieP Ring package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

**How to cite this package:** If you use the LieP Ring package, then please cite it as: *Bettina Eick and Michael Vaughan-Lee, LieP Ring – A GAP Package for computing with nilpotent Lie rings of prime-power order (2014), see*

<https://www.gap-system.org/Packages/liepring.html>

**Acknowledgements:** *The Lazard correspondence induces a one-to-one correspondence between the Lie  $p$ -rings of order  $p^n$  and class less than  $p$  and the  $p$ -groups of order  $p^n$  and class less than  $p$ . This package provides a function to evaluate this correspondence; this function has been implemented and given to us by Willem de Graaf.*

# 2

## Lie p-rings

*In this preliminary chapter we recall some of theoretic background of Lie rings and Lie p-rings. We refer to Chapter 5 in [Khu98] for some further details. Throughout we assume that  $p$  stands for a rational prime.*

*A Lie ring  $L$  is an additive abelian group with a multiplication that is alternating, bilinear and satisfies the Jacobi identity. We denote the product of two elements  $g$  and  $h$  of  $L$  with  $gh$ .*

*A subset  $I \subseteq L$  is an ideal in the Lie ring  $L$  if it is a subgroup of the additive group of  $L$  and it satisfies  $al \in I$  for all  $a \in I$  and  $l \in L$ . As the multiplication in  $L$  is alternating, it follows that  $la \in I$  for all  $l \in L$  and  $a \in I$ . Note that if  $I$  and  $J$  are ideals in  $L$ , then  $I + J = \{a + b \mid a \in I, b \in J\}$  and  $IJ = \langle ab \mid a \in I, b \in J \rangle_+$  are ideals in  $L$ .*

*A subset  $U \subseteq L$  is a subring of the Lie ring  $L$  if  $U$  is a Lie ring with respect to the addition and the multiplication of  $L$ . Every ideal in  $L$  is also a subring of  $L$ . As usual, for an ideal  $I$  in  $L$  the quotient  $L/I$  has the structure of a Lie ring, but this does not hold for subrings.*

*The lower central series of the Lie ring  $L$  is the series of ideals  $L = \gamma_1(L) \geq \gamma_2(L) \geq \dots$  defined by  $\gamma_i(L) = \gamma_{i-1}(L)L$ . We say that  $L$  is nilpotent if there exists a natural number  $c$  with  $\gamma_{c+1}(L) = \{0\}$ . The smallest natural number with this property is the class of  $L$ .*

*The notion of nilpotence now allows to state the central definition of this package. A **Lie p-ring** is a Lie ring that is nilpotent and has  $p^n$  elements for some natural number  $n$ .*

*Every finite dimensional Lie algebra over a field with  $p$  elements is an example for a Lie ring with  $p^n$  elements. Note that there exist non-nilpotent Lie algebras of this type: the Lie algebra consisting of all  $n \times n$  matrices with trace 0 and  $n \geq 3$  is an example. Thus not every Lie ring with  $p^n$  elements is nilpotent. (In contrast to the group case, where every group with  $p^n$  elements is nilpotent!)*

*For a Lie p-ring  $L$  we define the series  $L = \lambda_1(L) \geq \lambda_2(L) \geq \dots$  via  $\lambda_{i+1}(L) = \lambda_i(L)L + p\lambda_i(L)$ . This series is the lower exponent- $p$  central series of  $L$ . Its length is the  $p$ -class of  $L$ . If  $|L/\lambda_2(L)| = p^d$ , then  $d$  is the minimal generator number of  $L$ . Similar to the  $p$ -group case, one can observe that this is indeed the cardinality of a generating set of smallest possible size.*

*Each Lie p-ring  $L$  has a central series  $L = L_1 \geq \dots \geq L_n \geq \{0\}$  with quotients of order  $p$ . Choose  $l_i \in L_i \setminus L_{i+1}$  for  $1 \leq i \leq n$ . Then  $(l_1, \dots, l_n)$  is a generating set of  $L$  satisfying that  $pl_i \in L_{i+1}$  and  $l_i l_j \in L_{i+1}$  for  $1 \leq j < i \leq n$ . We call such a generating sequence a basis for  $L$  and we say that  $L$  has dimension  $n$ .*

# 3

# LiePRings in GAP

This package introduces a new datastructure that allows to define and compute with Lie  $p$ -rings in GAP. We first describe this datastructure in the case of ordinary Lie  $p$ -rings; that is, Lie  $p$ -rings for a fixed prime  $p$  with given structure constants. Then we show how this datastructure can also be used to define so-called 'generic' Lie  $p$ -rings; that is, Lie  $p$ -rings with indeterminate prime  $p$ .

## 3.1 Ordinary Lie $p$ -rings

Let  $p$  be a prime and let  $L$  be a Lie  $p$ -ring of order  $p^n$ . Let  $(l_1, \dots, l_n)$  be a basis for  $L$ . Then there exist coefficients  $c_{i,j,k} \in \{0, \dots, p-1\}$  so that the following relations hold in  $L$  for  $1 \leq i, j \leq n$  with  $i \neq j$ :

$$l_i \cdot l_j = \sum_{k=i+1}^n c_{i,j,k} l_k,$$

$$p l_i = \sum_{k=i+1}^n c_{i,i,k} l_k.$$

These structure constants define the Lie  $p$ -ring  $L$ . As the multiplication in a Lie  $p$ -ring is anticommutative, it follows that  $c_{i,j,k} = -c_{j,i,k}$  holds for each  $k$  and each  $i \neq j$ . Thus the structure constants  $c_{i,j,k}$  for  $i \geq j$  are sufficient to define the Lie  $p$ -ring  $L$ .

This package contains the new datastructure `LiePRing` that allows to define Lie  $p$ -rings via their structure constants  $c_{i,j,k}$ . To use this datastructure, we first collect all relevant information into a record as follows:

`dim`

the dimension  $n$  of  $L$ ;

`prime`

the prime  $p$  of  $L$ ;

`tab`

a list with structure constants  $[c_{1,1}, c_{2,1}, c_{2,2}, c_{3,1}, c_{3,2}, c_{3,3}, \dots]$ .

Each entry  $c_{i,j}$  in the list `tab` is a list  $[k_1, c_{i,j,k_1}, k_2, c_{i,j,k_2}, \dots]$  so that  $k_1 < k_2 < \dots$  and the entries  $c_{i,j,k_1}, c_{i,j,k_2}, \dots$  are the non-zero structure constants in the product  $l_i \cdot l_j$ . Thus if  $l_i \cdot l_j = 0$ , then  $c_{i,j}$  is the empty list. If an entry in the list `tab` is not bound, then it is assumed to be the empty list.

- 1 ► `LiePRingBySCTable( SC )`
- `LiePRingBySCTableNC( SC )`

These functions create a `LiePRing` from the structure constants table record `SC`. The first version checks that the multiplication defined by `tab` is alternating and satisfies the Jacobi-identity, the second version assumes that this is the case and omits these checks. These checks can also be carried out independently via the following function.

- 2 ► `CheckIsLiePRing( L )`

This function takes as input an object  $L$  created via `LiePRingBySCTableNC` and checks that the Jacobi identity holds in this ring.

The following example creates the Lie 2-ring of order 8 with trivial multiplication.

```
gap> SC := rec( dim := 3, prime := 2, tab := [] );;
gap> L := LiePRingBySCTable(SC);
<LiePRing of dimension 3 over prime 2>
gap> l := BasisOfLiePRing(L);
[ 11, 12, 13 ]
gap> l[1]*l[2];
0
gap> 2*l[1];
0
gap> l[1] + l[2];
11 + 12
```

The next example creates a LiePRing of order  $5^4$  with non-trivial multiplication.

```
gap> SC := rec( dim := 4, prime := 5, tab := [ [], [3, 1], [], [4, 1]] );;
gap> L := LiePRingBySCTableNC(SC);;
gap> ViewPCPresentation(L);
[12,11] = 13
[13,11] = 14
```

## 3.2 Generic Lie p-rings

In a generic Lie  $p$ -ring,  $p$  is allowed to be an indeterminate and the structure constants are allowed to be rational functions over a polynomial ring in a finite set of commuting indeterminates. It is generally assumed that the indeterminate with name  $p$  represents the prime, the indeterminate with name  $w$  represents the smallest primitive root modulo the prime and there are further predefined indeterminates with the names  $x, y, z, t, j, k, m, n, r, s, u$  and  $v$ . These indeterminates are used in the database of Lie  $p$ -rings and they can be obtained via

1 ► `IndeterminateByName( string )`

The structure constants records for generic Lie  $p$ -rings are similar to those for ordinary Lie  $p$ -rings, but have the additional entry `param` which is a list containing all indeterminates used in the considered Lie  $p$ -ring. We exhibit an example.

```
gap> p := IndeterminateByName("p");;
gap> x := IndeterminateByName("x");;
gap> S := rec( dim := 5,
>             param := [ x ],
>             prime := p,
>             tab := [ [ 4, 1 ], [ 3, 1 ], [ 5, x ], [ 4, 1 ], [ 5, 1 ] ] );;
gap> L := LiePRingBySCTable(S);
<LiePRing of dimension 5 over prime p with parameters [ x ]>
gap> ViewPCPresentation(L);
p*11 = 14
p*12 = x*15
[12,11] = 13
[13,11] = 14
[13,12] = 15
gap> l := BasisOfLiePRing(L);
[ 11, 12, 13, 14, 15 ]
gap> p*l[1];
14
```

```
gap> l[1]+l[2];
l1 + l2
gap> l[1]*l[2];
-1*l3
```

### 3.3 Specialising Lie $p$ -rings

A generic Lie  $p$ -ring defines a family of ordinary Lie  $p$ -rings by evaluating the parameters contained in its presentation. It is generally assumed that the indeterminate  $p$  is evaluated to a rational prime  $P$  and the indeterminate  $w$  is evaluated to the smallest primitive root modulo  $P$  (this can be determined via `PrimitiveRootMod(P)`). All other indeterminates can take arbitrary integer values (usually these values are in  $\{0, \dots, P-1\}$ , but other choices are possible as well). The following functions allow to evaluate the indeterminates.

#### 1 ► `SpecialiseLiePRing(L, P, para, vals)`

takes as input a generic Lie  $p$ -ring  $L$ , a rational prime  $P$ , a list of indeterminates  $para$  and a corresponding list of values  $vals$ . The function returns a new Lie  $p$ -ring in which the prime  $p$  is evaluated to  $P$ , the parameter  $w$  is evaluated to `PrimitiveRootMod(P)` and the parameters in  $para$  are evaluated to  $vals$ .

#### 2 ► `SpecialisePrimeOfLiePRing(L, P)`

this is a shortcut for `SpecialiseLiePRing(L, P, [], [])`. We exhibit a some example applications.

```
gap> p := IndeterminateByName("p");;
gap> w := IndeterminateByName("w");;
gap> x := IndeterminateByName("x");;
gap> y := IndeterminateByName("y");;
gap> S := rec( dim := 7,
>             param := [ w, x, y ],
>             prime := p,
>             tab := [ [ ], [ 6, 1 ], [ 6, 1 ], [ 7, 1 ], [ ],
>                      [ 6, x, 7, y ], [ ], [ 7, 1 ], [ 6, w ] ] );;
gap> L := LiePRingBySCTable(S);
<LiePRing of dimension 7 over prime p with parameters [ w, x, y ]>
gap> ViewPCPresentation(L);
p*l2 = l6
p*l3 = x*l6 + y*l7
[l2,l1] = l6
[l3,l1] = l7
[l4,l2] = l7
[l4,l3] = w*l6
gap> SpecialiseLiePRing(L, 7, [x, y], [0,0]);
<LiePRing of dimension 7 over prime 7>
gap> ViewPCPresentation(last);
7*l2 = l6
[l2,l1] = l6
[l3,l1] = l7
[l4,l2] = l7
[l4,l3] = 3*l6
gap> SpecialiseLiePRing(L, 11, [x, y], [0,10]);
<LiePRing of dimension 7 over prime 11>
gap> ViewPCPresentation(last);
11*l2 = l6
11*l3 = 10*l7
```

```

[12,11] = 16
[13,11] = 17
[14,12] = 17
[14,13] = 2*16
gap> Cartesian([0,1],[0,1]);
[ [ 0, 0 ], [ 0, 1 ], [ 1, 0 ], [ 1, 1 ] ]
gap> List(last, v -> SpecialiseLiePRing(L, 2, [x,y], v));
[ <LiePRing of dimension 7 over prime 2>,
  <LiePRing of dimension 7 over prime 2>,
  <LiePRing of dimension 7 over prime 2>,
  <LiePRing of dimension 7 over prime 2> ]

```

*It is not necessary to specialise all parameters at once. In particular, it is possible to leave the prime  $p$  as indeterminate and specialize only some of the parameters. (Except for  $w$  which is linked to  $p$ .)*

```

gap> SpecialiseLiePRing(L, p, [x], [0]);
<LiePRing of dimension 7 over prime p with parameters [ y, w ]>
gap> ViewPCPresentation(last);
p*12 = 16
p*13 = y*17
[12,11] = 16
[13,11] = 17
[14,12] = 17
[14,13] = w*16
gap> SpecialiseLiePRing(L, p, [y], [3]);
<LiePRing of dimension 7 over prime p with parameters [ x, w ]>
gap> ViewPCPresentation(last);
p*12 = 16
p*13 = x*16 + 3*17
[12,11] = 16
[13,11] = 17
[14,12] = 17
[14,13] = w*16

```

*It is also possible to specialise the prime only, but leave all or some of the parameters indeterminate. Note that specialising  $p$  also specialises  $w$ . Again, we continue to use the generic Lie  $p$ -ring  $L$  as above.*

```

gap> SpecialisePrimeOfLiePRing(L, 29);
<LiePRing of dimension 7 over prime 29 with parameters [ y, x ]>
gap> ViewPCPresentation(last);
29*12 = 16
29*13 = x*16 + y*17
[12,11] = 16
[13,11] = 17
[14,12] = 17
[14,13] = 2*16

```

### 3 ► LiePValues(K)

*if  $K$  is obtained by specialising, then this attribute is set and contains the parameters that have been specialised and their values.*



```

gap> L := LiePRingsByLibrary(6)[14];
<LiePRing of dimension 6 over prime p with parameters [ x ]>
gap> K := SpecialisePrimeOfLiePRing(L, 5);
<LiePRing of dimension 6 over prime 5 with parameters [ x ]>
gap> LiePValues(K);
[ [ p, w ], [ 5, 2 ] ]

```

### 3.4 Subrings of Lie $p$ -rings

Let  $L$  be a Lie  $p$ -ring with basis  $(l_1, \dots, l_n)$  and let  $U$  be a subring of  $L$ . Then  $U$  is a Lie  $p$ -ring and thus also has a basis  $(u_1, \dots, u_m)$ . For  $1 \leq i \leq m$  we define the coefficients  $a_{ij} \in \{0, \dots, p-1\}$  via

$$u_i = \sum_{j=1}^n a_{ij} l_j$$

and we denote with  $A$  the matrix with entries  $a_{ij}$ . We say that the basis  $(u_1, \dots, u_m)$  is induced if  $A$  is in upper triangular form. Further, the basis  $(u_1, \dots, u_m)$  is canonical if  $A$  is in upper echelon form; that is, it is upper triangular, each row in  $A$  has leading entry 1 and there are 0's above the leading entry. Note that a canonical basis is unique for the subring.

#### 1 ► LiePSubring(L, gens)

Let  $L$  be a (generic or ordinary) Lie  $p$ -ring and let  $gens$  be a set of elements in  $L$ . This function determines a canonical basis for the subring generated by  $gens$  in  $L$  and returns the LiePSubring of  $L$  generated by  $gens$ . Note that this function may have strange effects for generic Lie  $p$ -rings as the following example shows.

```

gap> L := LiePRingsByLibrary(6)[100];
<LiePRing of dimension 6 over prime p>
gap> l := BasisOfLiePRing(L);
[ l1, l2, l3, l4, l5, l6 ]
gap> U := LiePSubring(L, [5*l[1]]);
<LiePRing of dimension 3 over prime p>
gap> BasisOfLiePRing(U);
[ l1, l4, l6 ]
gap> K := SpecialisePrimeOfLiePRing(L, 5);
<LiePRing of dimension 6 over prime 5>
gap> b := BasisOfLiePRing(K);
[ b1, b2, b3, b4, b5, b6 ]
gap> LiePSubring(K, [5*b[1]]);
<LiePRing of dimension 2 over prime 5>
gap> BasisOfLiePRing(last);
[ b4, b6 ]
gap> K := SpecialisePrimeOfLiePRing(L, 7);
<LiePRing of dimension 6 over prime 7>
gap> b := BasisOfLiePRing(K);
[ b1, b2, b3, b4, b5, b6 ]
gap> U := LiePSubring(K, [5*b[1]]);
<LiePRing of dimension 3 over prime 7>
gap> BasisOfLiePRing(U);
[ b1, b4, b6 ]

```

#### 2 ► LiePIdeal(L, gens)

return the ideal of  $L$  generated by  $gens$ . This function computes a an induced basis for the ideal.

```
gap> LiePIdeal(L, [1[1]]);
<LieP Ring of dimension 5 over prime p>
gap> BasisOfLieP Ring(last);
[ 11, 13, 14, 15, 16 ]
```

### 3 ► LiePQuotient(L, U)

*return a Lie p-ring isomorphic to  $L/U$  where  $U$  must be an ideal of  $L$ . This function requires that  $L$  is an ordinary Lie p-ring.*

```
gap> LiePIdeal(K, [b[1]]);
<LieP Ring of dimension 5 over prime 7>
gap> LiePIdeal(K, [b[2]]);
<LieP Ring of dimension 4 over prime 7>
gap> LiePQuotient(K,last);
<LieP Ring of dimension 2 over prime 7>
```

## 3.5 Elementary functions

*The functions described in this section work for ordinary and generic Lie p-rings and their subrings.*

### 1 ► PrimeOfLieP Ring(L)

*returns the underlying prime. This can either be an integer or an indeterminate.*

### 2 ► BasisOfLieP Ring(L)

*returns a basis for  $L$ .*

### 3 ► DimensionOfLieP Ring(L)

*returns the dimension of  $L$ .*

### 4 ► ParametersOfLieP Ring(L)

*returns the list of indeterminates involved in  $L$ . If  $L$  is a subring of a Lie p-ring defined by structure constants, then the parameters of the parent are returned.*

### 5 ► ViewPCPresentation(L)

*prints the presentation for  $L$  with respect to its basis.*

## 3.6 Series of subrings

*Let  $L$  be a generic or ordinary Lie p-ring or a subring of such a Lie p-ring.*

### 1 ► LiePLowerCentralSeries(L)

*returns the lower central series of  $L$ .*

### 2 ► LiePLowerPCentralSeries(L)

*returns the lower exponent-p central series of  $L$ .*

### 3 ► LiePDerivedSeries(L)

*returns the derived series of  $L$ .*

### 4 ► LiePMinimalGeneratingSet(L)

*returns a minimal generating set of  $L$ ; that is, a generating set of smallest possible size.*

### 3.7 The Lazard correspondence

*The following function has been implemented by Willem de Graaf. It uses the Baker-Campbell-Hausdorff formula as described in [\[CdGVL12\]](#) and it is based on the Liering package [\[CdG10\]](#).*

1 ► PGroupByLiePRing(L)

*Let  $L$  be an ordinary Lie  $p$ -ring with  $\text{cl}(L) < p$ . Then this function returns the  $p$ -group  $G$  obtained from  $L$  via the Lazard correspondence.*

# The Database

For each  $n \in \{1, \dots, 7\}$  this package contains a (finite) list of generic presentations of Lie  $p$ -rings. For each prime  $p \geq 5$ , each of the generic Lie  $p$ -rings gives rise to a family of Lie  $p$ -rings over the considered prime  $p$  by specialising the indeterminates to a certain list of values. The resulting lists of Lie  $p$ -rings provides a complete and irredundant set of isomorphism type representatives of the Lie  $p$ -rings of order  $p^n$ . The generic Lie  $p$ -rings of  $p$ -class at most 2 can also be considered for the prime  $p = 3$  and yield a list of isomorphism type representatives for the Lie  $p$ -rings of order  $3^n$  and  $p$ -class at most 2.

In the following we describe functions to access the database. Throughout this chapter, we assume that  $\text{dim} \in \{1, \dots, 7\}$  and  $P$  is a prime with  $P \neq 2$ .

```
1 ▶ LiePRingsByLibrary( dim )
  ▶ LiePRingsByLibrary( dim, gen, cl )
```

```
2 ▶ LiePRingsByLibrary( dim, P )
  ▶ LiePRingsByLibrary( dim, P, gen, cl )
```

The first example yields the generic Lie  $p$ -rings of dimension 4.

[illegible]

The next example yields the isomorphism type representatives of Lie  $p$ -rings of dimension 3 for the prime 5.

The following example extracts the generic Lie  $p$ -rings of dimension 5 with minimal generator number 2 and  $p$ -class 4.

Finally, we determine the isomorphism type representatives of Lie  $p$ -rings of dimension 5, minimal generator number 2 and  $p$ -class 4 for the prime 7.

[illegible]

## 4.2 Numbers of Lie p-rings

### 1 ► NumberOfLiePRings( dim )

returns the number of generic Lie p-rings in the database of the considered dimension for  $\dim\{1, \dots, 7\}$ .

```
gap> List([1..7], x -> NumberOfLiePRings(x));
[ 1, 2, 5, 15, 75, 542, 4773 ]
```

### 2 ► NumberOfLiePRings( dim, P )

returns the number of isomorphism types of ordinary Lie p-rings of order  $P^{\dim}$  in the database. If  $P \geq 5$ , then this is the number of all isomorphism types of Lie p-rings of order  $P^{\dim}$  and if  $P = 3$  then this is the number of all isomorphism types of Lie p-rings of p-class at most 2. If  $P \geq 7$ , then this number coincides with  $\text{NumberSmallGroups}(P^{\dim})$ .

### 3 ► NumberOfLiePRingsInFamily( L )

returns the number of Lie p-rings associated to  $L$  as a polynomial in  $p$  and possibly some residue classes.

```
gap> L := LiePRingsByLibrary(7)[780];
<LiePRing of dimension 7 over prime p with parameters
[ x, y, z, t, s, u, v ]>
gap> NumberOfLiePRingsInFamily(L);
-1/3*p^5*(p-1,3)+p^5-1/3*p^4*(p-1,3)+p^4-1/3*p^3*(p-1,3)+p^3-1/3*p^2*(p-1,3)
+p^2-p*(p-1,3)+3*p-3/2*(p-1,3)+9/2
```

## 4.3 Searching the database

We now consider a generic Lie p-ring  $L$  from the database and consider the family of ordinary Lie p-rings that arise from it.

### 1 ► LiePRingsInFamily( L, P )

takes as input a generic Lie p-ring  $L$  from the database and a prime  $P$  and returns all Lie p-rings determined by  $L$  and  $P$  up to isomorphism. This function returns fail if the generic Lie p-ring does not exist for the special prime  $P$ ; this may be due to the conditions on the prime or (if  $P = 3$ ) to the p-class of the Lie p-ring.

```
gap> L := LiePRingsByLibrary(7)[118];
<LiePRing of dimension 7 over prime p with parameters [ x, y ]>
gap> LibraryConditions(L);
[ "[x,y]^[x,-y]", "p=1 mod 4" ]
gap> LiePRingsInFamily(L, 7);
fail
gap> Length(LiePRingsInFamily(L,13));
91
gap> 13^2;
169
```

The following example shows how to determine all Lie p-rings of dimension 5 and p-class 4 over the prime 29 up to isomorphism.

```

gap> L := LiePRingsByLibrary(5);;
gap> L := Filtered(L, x -> PClassOfLiePRing(x)=4);
[ <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p>,
  <LiePRing of dimension 5 over prime p> ]
gap> K := List(L, x-> LiePRingsInFamily(x, 29));
[ [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ], fail, fail,
  [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ], fail, fail,
  [ <LiePRing of dimension 5 over prime 29> ],
  [ <LiePRing of dimension 5 over prime 29> ] ]
gap> K := Filtered(Flat(K), x -> x<>fail);
[ <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29>,
  <LiePRing of dimension 5 over prime 29> ]

```

## 4.4 More details

Let  $L$  be a Lie  $p$ -ring from the database. Then the following additional attributes are available.

1 ► **LibraryName( $L$ )**

returns a string with the name of  $L$  in the database. See [p567.pdf](#) for further background.

2 ► **ShortPresentation( $L$ )**

returns a string exhibiting a short presentation of  $L$ .

3 ► `LibraryConditions(L)`

*returns the conditions on L. This is a list of two strings. The first string exhibits the conditions on the parameters of L, the second shows the conditions on primes.*

4 ► `MinimalGeneratorNumberOfLieP Ring(L)`

*returns the minimal generator number of L.*

5 ► `PClassOfLieP Ring(L)`

*returns the p-class of L.*

```
gap> L := LiePRingsByLibrary(7)[118];
<LieP Ring of dimension 7 over prime p with parameters [ x, y ]>
gap> LibraryName(L);
"7.118"
gap> LibraryConditions(L);
[ "[x,y]^[x,-y]", "p=1 mod 4" ]
```

*All of the information listed in this section is inherited when L is specialised.*

```
gap> L := LiePRingsByLibrary(7)[118];
<LieP Ring of dimension 7 over prime p with parameters [ x, y ]>
gap> K := SpecialiseLieP Ring(L, 13, ParametersOfLieP Ring(L), [0,0]);
<LieP Ring of dimension 7 over prime 13>
gap> LibraryName(K);
"7.118"
gap> LibraryConditions(K);
[ "[x,y]^[x,-y]", "p=1 mod 4" ]
```

*The following example shows how to find a Lie p-ring with a given name in the database.*

```
gap> L := LiePRingsByLibrary(7);;
gap> Filtered(L, x -> LibraryName(x) = "7.1010")[1];
<LieP Ring of dimension 7 over prime p>
```

## 4.5 Special functions for dimension 7

*The database of Lie p-rings of dimension 7 is very large and it may be time-consuming (or even impossible due to storage problems) to generate all Lie p-rings of dimension 7 for a given prime P.*

*Thus there are some special functions available that can be used to access a particular set of Lie p-rings of dimension 7 only. In particular, it is possible to consider the descendants of a single Lie p-ring of smaller dimension by itself. The Lie p-rings of this type are all stored in one file of the library. Thus, equivalently, it is possible to access the Lie p-rings in one single file only.*

*The table LIE\_TABLE contains a list of all possible files together with the number of Lie p-rings generated by their corresponding Lie p-rings.*

1 ► `LiePRingsDim7ByFile( nr )`

*returns the generic Lie p-rings in file number nr.*

2 ► `LiePRingsDim7ByFile( nr, P )`

*returns the isomorphism types of Lie p-rings in file number nr for the prime P.*



```

gap> LIE_TABLE[100];
[ "3gen/gapdec6.139", 1/2*p+(p-1,3)+3/2 ]
gap> LiePRingsDim7ByFile(100);
[ <LiePRing of dimension 7 over prime p>,
  <LiePRing of dimension 7 over prime p>,
  <LiePRing of dimension 7 over prime p>,
  <LiePRing of dimension 7 over prime p>,
  <LiePRing of dimension 7 over prime p with parameters [ x ]> ]
gap> LiePRingsDim7ByFile(100, 7);
[ <LiePRing of dimension 7 over prime 7>,
  <LiePRing of dimension 7 over prime 7>,
  <LiePRing of dimension 7 over prime 7>,
  <LiePRing of dimension 7 over prime 7>,
  <LiePRing of dimension 7 over prime 7>,
  <LiePRing of dimension 7 over prime 7>,
  <LiePRing of dimension 7 over prime 7>,
  <LiePRing of dimension 7 over prime 7> ]

```

## 4.6 Dimension 8 and maximal class

Recently, Lee and Vaughan-Lee [\[LVL22\]](#) determined the Lie  $p$ -rings of dimension 8 with maximal class up to isomorphism. This classification is now also available in the Lie  $p$ -ring package via the following functions.

1 ► `LiePRingsByLibraryMC8()`

returns a list of 69 generic Lie  $p$ -rings. For each of these the following function returns the isomorphism types of Lie  $p$ -rings in the family for a fixed prime  $P$  with  $P \geq 5$ .

2 ► `LiePRingsInFamilyMC8(L, P)`

# 5 Advanced functions for Lie p-rings

*This chapter described a few more advanced functions available for generic Lie p-rings.*

## 5.1 Schur multipliers

*The package contains a method to determine the Schur multipliers of the Lie p-rings in the family defined by a generic Lie p-ring.*

1 ► `LiePSchurMult( L )`

*The function takes as input a generic Lie p-ring and determines a list of possible Schur multipliers, each described by its abelian invariants, for the Lie p-rings in the family described by L. For each entry in the list of Schur multipliers there is a description of those parameters which give the considered entry. This description consists of two lists 'units' and 'zeros'. Both consist of rational functions over the parameters of the Lie p-ring. The parameters described by these lists are which evaluate to zero for each rational function in 'zeros' and evaluate not to zero for each rational function in 'units'.*

```
gap> LL := LiePRingsByLibrary(7);
gap> L := Filtered(LL, x -> Length(ParametersOfLiePRing(x))=2)[1];
<LiePRing of dimension 7 over prime p with parameters [ x, y ]>
gap> NumberOfLiePRingsInFamily(L);
p^2-p
gap> RingInvariants(L);
rec( units := [ x ], zeros := [ ] )
gap> ss := LiePSchurMult(L);
[ rec( norm := [ p ], units := [ x, y ], zeros := [ x*y^2-x*y+1 ] ),
  rec( norm := [ p^2 ], units := [ x ], zeros := [ x*y ] ),
  rec( norm := [ p ], units := [ x, x*y^2-x*y+1, y ], zeros := [ ] ) ]
```

*In this example, L defines a generic Lie p-rings with two parameters and the RingInvariants of L show that the parameter x should be non-zero. The function LiePSchurMult(L) yields that there are two possible Schur multipliers for the Lie p-rings in the family defined by L: the cyclic groups of order p and of order  $p^2$ . The second option only arises if  $xy = 0$  and thus, as x is non-zero, if  $y = 0$ .*

*The package also contains a function that tries to determine the numbers of values of the parameters satisfying the conditions of a description of a Schur multiplier. This succeeds in many cases and returns a polynomial in p in this case. If it does not succeed then it returns fail.*

2 ► `ElementNumbers( pp, ss )`

*We continue the above example.*

```
gap> ElementNumbers(ParametersOfLiePRing(L), ss);
rec( norms := [ [ p^2 ], [ p ] ], numbs := [ p-1, p^2-2*p+1 ] )
```

## 5.2 Automorphism groups

The package contains a function that determines a description for the automorphism groups of the Lie  $p$ -rings in the family defined by a generic Lie  $p$ -ring.

1 ► `AutGrpDescription( L )`

Each automorphism of  $L$  is defined by its images on a generating set of  $L$ . If  $l_1, \dots, l_n$  is a basis of  $L$  and  $l_1, \dots, l_d$  is a generating set, then each automorphism is defined by the images of  $l_1, \dots, l_d$  and each image is an integral linear combination of the basis elements  $l_1, \dots, l_n$ . The function `AutGrpDescription` returns a matrix containing a description of the coefficients in each linear combination and a list of relations among these coefficients. We consider two examples.

```
gap> L := Filtered(LL, x -> Length(ParametersOfLiePRing(x))=2)[1];
<LiePRing of dimension 7 over prime p with parameters [ x, y ]>
gap> AutGroupDescription(L);
rec( auto := [ [ 1, 0, A13, A14, A15, A16, A17 ],
               [ 0, 1, A23, A24, A25, A26, A27 ] ],
      eqns := [ [ ], [ ] ] )
gap> L := Filtered(LL, x -> Length(ParametersOfLiePRing(x))=2)[2];
<LiePRing of dimension 7 over prime p with parameters [ x, y ]>
gap> AutGroupDescription(L);
rec( auto := [ [ A22^3, 0, A13, A14, A15, A16, A17 ],
               [ 0, A22, A23, A24, A25, A26, A27 ] ],
      eqns := [ [ A22*A24-1/2*A23^2, A22^2*y-y,
                  A22*A23^2*y-2*A24*y, A22^4-1,
                  A23^4*y-4*A24^2*y, A22^3*A23^2-2*A24,
                  A22^2*A23^4-4*A24^2, A22*A23^6-8*A24^3,
                  A23^8-16*A24^4 ] ] ] )
```

In both cases,  $L$  is generated by the first two entries in its basis and hence the automorphism group matrix has two rows and seven columns. In the first case,  $L$  has  $p^{10}$  automorphisms inducing the identity on the Frattini-quotient of  $L$ . In the second case, the automorphism group matrix shows that each automorphism induces a certain type of diagonal matrix on the Frattini-quotient of  $L$  and there are further equations among the coefficients of the matrix. These further equations are equivalent to  $A22^2 = 1$  and  $A24 = A22A23^2/2$ . Hence  $L$  has  $2p^9$  automorphisms.

The entry `eqns` is a list of lists. The equations in the  $i$ th entry of this list have to be satisfied mod  $p^i$ .

In a few special cases, the function returns a list of possible automorphisms together with related equations and conditions. We exhibit an example.

```
gap> L := LiePRingsByLibrary(7)[489];
<LiePRing of dimension 7 over prime p with parameters [ x ]>
gap> AutGroupDescription(L);
[ rec( auto := [ [ 1, 0, A13, A14, A15, A16, A17 ],
                 [ 0, 1, A23, A24, A25, A26, A27 ] ],
        comment := "p^8 automorphisms",
        eqns := [ [ A13^2*x-A13*A23+2*A15*x+A14-A25,
                    -A13*A23*x+A14*x+A23^2-A25*x-2*A24 ] ] ),
  rec( auto := [ [ 0, A12, A13, A14, A15, A16, A17 ],
                 [ -x, 0, A23, A24, A25, A26, A27 ] ],
        comment := "p^8 automorphisms when x <> 0 mod p",
        eqns := [ [ A12^2*A24*x-A12*A13*A23*x+A12*A13*x^2
                    +2*A12*A15*x^2+A12*A14*x-A13^2*x+A13*x+A15*x-A14,
                    -A12^2*A23*x^3+A12*A13*x^3+A12*A23^2*x-A12*A25*x^2
                    -2*A12*A24*x+A13*A23*x+A13*x^2-A15*x^2+A23*x+A25*x-A24 ] ],
                 [ A12*x+1 ] ] ) ]
```

*In this example  $A12x = -1$  modulo  $p^2$ . We note that different choices for  $A12$  do not give different automorphisms. Hence a single solution for  $A12$  is sufficient to describe all automorphisms.*

# Bibliography

- [CdG10] *Serena Cicalò and Willem A. de Graaf. Liering, 2010. A GAP 4 package.*
- [CdGVL12] *Serena Cicalò, Willem A. de Graaf, and Michael R. Vaughan-Lee. An effective version of the Lazard correspondence. J. Algebra, 352:430–450, 2012.*
- [Khu98] *E. I. Khukhro.  $p$ -automorphisms of finite  $p$ -groups, volume 246 of London Mathematical Society Lecture Note Series. Cambridge University Press, Cambridge, 1998.*
- [LVL22] *Seungjai Lee and Michael Vaughan-Lee. The groups and nilpotent lie rings of order  $p^8$  and maximal class. To appear, 2022.*
- [NOVL03] *Mike F. Newman, Eamonn A. O’Brien, and Michael R. Vaughan-Lee. Groups and nilpotent Lie rings whose order is the sixth power of a prime. J. Alg., 278:383 – 401, 2003.*
- [OVL05] *Eamonn A. O’Brien and Michael R. Vaughan-Lee. The groups with order  $p^7$  for odd prime  $p$ . J. Algebra, 292(1):243–258, 2005.*
- [VL13] *Michael R. Vaughan-Lee. The nilpotent Lie rings of order  $p^k$  for  $k \leq 7$ , 2013. See within this package under /lib/notes/p567.pdf.*

# 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

*Accessing Lie p-rings, 12*  
*AutGrpDescription, 19*  
*Automorphism groups, 19*

## B

*BasisOfLiePRing, 10*

## C

*CheckIsLiePRing, 5*

## D

*Dimension 8 and maximal class, 17*  
*DimensionOfLiePRing, 10*

## E

*Elementary functions, 10*  
*ElementNumbers, 18*

## G

*Generic Lie p-rings, 6*

## I

*IndeterminateByName, 6*

## L

*LibraryConditions, 16*  
*LibraryName, 15*  
*LiePDerivedSeries, 10*  
*LiePIdeal, 9*  
*LiePLowerCentralSeries, 10*  
*LiePLowerPCentralSeries, 10*  
*LiePMinimalGeneratingSet, 10*  
*LiePQuotient, 10*  
*LiePRingBySCTable, 5*  
*LiePRingBySCTableNC, 5*  
*LiePRingsByLibrary, 12*  
*LiePRingsByLibraryMC8, 17*  
*LiePRingsDim7ByFile, 16*  
*LiePRingsInFamily, 14*

*LiePRingsInFamilyMC8, 17*  
*LiePSchurMult, 18*  
*LiePSubring, 9*  
*LiePValues, 8*

## M

*MinimalGeneratorNumberOfLiePRing, 16*  
*More details, 15*

## N

*NumberOfLiePRings, 14*  
*NumberOfLiePRingsInFamily, 14*  
*Numbers of Lie p-rings, 14*

## O

*Ordinary Lie p-rings, 5*

## P

*ParametersOfLiePRing, 10*  
*PClassOfLiePRing, 16*  
*PGroupByLiePRing, 11*  
*PrimeOfLiePRing, 10*

## S

*Schur multipliers, 18*  
*Searching the database, 14*  
*Series of subrings, 10*  
*ShortPresentation, 15*  
*Special functions for dimension 7, 16*  
*SpecialiseLiePRing, 7*  
*SpecialisePrimeOfLiePRing, 7*  
*Specialising Lie p-rings, 7*  
*Subrings of Lie p-rings, 9*

## T

*The Lazard correspondence, 11*

## V

*ViewPCPresentation, 10*