Unipot

—

A GAP4 Package

For Computations with elements of unipotent subgroups of Chevalley Groups

Version 1.2

by

Sergei Haller
(Sergei.Haller@math.uni-giessen.de)

Arbeitsgruppe Algebra
Mathematisches Institut, Justus-Liebig-Universität Gießen
Arndtstr. 2, 35392 Gießen, Germany

Contents

1 Preface **3**

1.1 Root Systems . . . 3

1.2 Citing Unipot . . . 3

2 The GAP Package Unipot **4**

2.1 General functionality . . . 4

2.2 Unipotent subgroups of Chevalley groups . . . 4

2.3 Elements of unipotent subgroups of Chevalley groups . . . 7

2.4 Symbolic computation . . . 12

Bibliography **13**

Index **14**

1 Preface

Unipot is a package for GAP4 [GAP04]. The version 1.0 of this package was the content of my diploma thesis [Hal00].

Let U be a unipotent subgroup of a Chevalley group of Type $L(K)$. Then it is generated by the elements $x_r(t)$ for all $r \in \Phi^+, t \in K$. The roots of the underlying root system Φ are ordered according to the height function. Each element of U is a product of the root elements $x_r(t)$. By Theorem 5.3.3 from [Car89] each element of U can be uniquely written as a product of root elements with roots in increasing order. This unique form is called the canonical form.

The main purpose of this package is to compute the canonical form of an element of the group U. For we have implemented the unipotent subgroups of Chevalley groups and their elements as GAP objects and installed some operations for them. One method for the operation `Comm` uses the Chevalley's commutator formula, which we have implemented, too.

1.1 Root Systems

We are using the root systems and the structure constants available in GAP from the simple Lie algebras. We also are using the same ordering of roots available in GAP.

Note that the structure constants in GAP4.1 are not generated corresponding to a Chevalley basis, so computations in the groups of type B_l may produce an error and computations in groups of types B_l, C_l and F_4 may lead to wrong results. In the groups of other types we haven't seen any wrong results but can not guarantee that all results are correct.

Since the revision 4.2 of GAP the structure constants are generated corresponding to a Chevalley basis, so that they meet all our assumptions.

Therefore the package requires at least the revision 4.2 of GAP.

Beginning with version 1.2 of Unipot, the new package loading mechanism of GAP4.4 is used and therefore, GAP4.4 is required.

1.2 Citing Unipot

If you use Unipot to solve a problem or publish some result that was partly obtained using Unipot, I would appreciate it if you would cite Unipot, just as you would cite another paper that you used. (Below is a sample citation.) Again I would appreciate if you could inform me about such a paper.

Specifically, please refer to:

```
[Hal02] Sergei Haller. Unipot --- a system for computing with elements
        of unipotent subgroups of Chevalley groups, Version 1.2.
        Justus-Liebig-Universitaet Giessen, Germany, July 2002.
        (http://...)
```

(Should the reference style require full addresses please use: “Arbeitsgruppe Algebra, Mathematisches Institut, Justus-Liebig-Universität Gießen, Arndtstr. 2, 35392 Gießen, Germany”)

2 The GAP Package Unipot

This chapter describes the package Unipot. Mainly, the package provides the ability to compute with elements of unipotent subgroups of Chevalley groups, but also some properties of this groups.

In this chapter we will refer to unipotent subgroups of Chevalley groups as “unipotent subgroups” and to elements of unipotent subgroups as “unipotent elements”. Specifically, we only consider unipotent subgroups generated by all positive root elements.

2.1 General functionality

In this section we will describe the general functionality provided by this package.

1 ► `UnipotChevInfo` V

`UnipotChevInfo` is an `InfoClass` used in this package. `InfoLevel` of this `InfoClass` is set to 1 by default and can be changed to any level by `SetInfoLevel( UnipotChevInfo,` *n* `)`.

Following levels are used throughout the package:

1. —
2. When calculating the order of a finite unipotent subgroup, the power presentation of this number is printed. (See 2.2.5 for an example)
3. When comparing unipotent elements, output, for which of them the canonical form must be computed. (See 2.3.11 for an example)
4. —
5. While calculating the canonical form, output the different steps.
6. The process of calculating the Chevalley commutator constants is printed on the screen

2.2 Unipotent subgroups of Chevalley groups

In this section we will describe the functionality for unipotent subgroups provided by this package.

1 ► `IsUnipotChevSubGr(` *grp* `)` C

Category for unipotent subgroups.

2 ► `UnipotChevSubGr(` *type*`,` *n*`,` *F* `)` F

`UnipotChevSubGr` returns the unipotent subgroup U of the Chevalley group of type *type*, rank n over the ring F.

type must be one of `"A"`, `"B"`, `"C"`, `"D"`, `"E"`, `"F"`, `"G"`.

For the type `"A"`, n must be a positive integer.

For the types `"B"` and `"C"`, n must be a positive integer ≥ 2.

For the type `"D"`, n must be a positive integer ≥ 4.

For the type `"E"`, n must be one of $6, 7, 8$.

For the type `"F"`, n must be 4.

For the type `"G"`, n must be 2.

```
gap> U_G2 := UnipotChevSubGr("G", 2, Rationals);
<Unipotent subgroup of a Chevalley group of type G2 over Rationals>
gap> IsUnipotChevSubGr(U_G2);
true

gap> UnipotChevSubGr("E", 3, Rationals);
Error, <n> must be one of 6, 7, 8 for type E  called from
UnipotChevFamily( type, n, F ) called from
<function>( <arguments> ) called from read-eval-loop
Entering break read-eval-print loop ...
you can 'quit;' to quit to outer loop, or
you can 'return;' to continue
brk>
```

3 ► `PrintObj(` *U* `)` M
► `ViewObj(` *U* `)` M

Special methods for unipotent subgroups. (see GAP Reference Manual, section 6.2 for general information on `View` and `Print`)

```
gap> Print(U_G2);
UnipotChevSubGr( "G", 2, Rationals )gap> View(U_G2);
<Unipotent subgroup of a Chevalley group of type G2 over Rationals>gap>
```

4 ► `One(` *U* `)` M
► `OneOp(` *U* `)` M

Special methods for unipotent subgroups. Return the identity element of the group U. The returned element has representation `UNIPOT_DEFAULT_REP` (see 2.3.3).

5 ► `Size(` *U* `)` M

`Size` returns the order of a unipotent subgroup. This is a special method for unipotent subgroups using the result in Carter [Car89], Theorem 5.3.3 (ii).

```
gap> SetInfoLevel( UnipotChevInfo, 2 );
gap> Size( UnipotChevSubGr("E", 8, GF(7)) );
#I  The order of this group is 7^120 which is
25808621098934927604791781741317238363169114027609954791128059842592785343731\
7437263620645695945672001
gap> SetInfoLevel( UnipotChevInfo, 1 );
```

6 ► `RootSystem(` *U* `)` M

This method is similar to the method `RootSystem` for semisimple Lie algebras (see Section 61.7 in the GAP Reference Manual for further information).

`RootSystem` returns the underlying root system of the unipotent subgroup U. The returned object is from the category `IsRootSystem`:

```
gap> R_G2 := RootSystem(U_G2);
<root system of rank 2>
gap> IsRootSystem(last);
true
gap> SimpleSystem(R_G2);
[ [ 2, -1 ], [ -3, 2 ] ]
gap>
```

Additionally to the properties and attributes described in the Reference Manual, following attributes are installed for the Root Systems by the package Unipot:

7 ▶ `PositiveRootsFC(` *R* `)` A
▶ `NegativeRootsFC(` *R* `)` A

The list of positive resp. negative roots of the root system R. Every root is represented as a list of coefficients of the linear combination in fundamental roots. E.g. let $r = \sum_{i=1}^{l} k_i r_i$, where $r_1, \ldots, r_l$ are the fundamental roots, then r is represented as the list $[k_1, \ldots, k_l]$.

```
gap> U_E6 := UnipotChevSubGr("E",6,GF(2));
<Unipotent subgroup of a Chevalley group of type E6 over GF(2)>
gap> R_E6 := RootSystem(U_E6);
<root system of rank 6>
gap> PositiveRoots(R_E6){[1..6]};
[ [ 2,  0, -1, 0,  0, 0 ], [ 0, 2, 0, -1, 0,  0 ], [ -1, 0, 2, -1,  0, 0 ],
  [ 0, -1, -1, 2, -1, 0 ], [ 0, 0, 0, -1, 2, -1 ], [  0, 0, 0,  0, -1, 2 ] ]
gap> PositiveRootsFC(R_E6){[1..6]};
[ [ 1, 0, 0, 0, 0, 0 ], [ 0, 1, 0, 0, 0, 0 ], [ 0, 0, 1, 0, 0, 0 ],
  [ 0, 0, 0, 1, 0, 0 ], [ 0, 0, 0, 0, 1, 0 ], [ 0, 0, 0, 0, 0, 1 ] ]
gap>
gap> PositiveRootsFC(R)[Length(PositiveRootsFC(R_E6))]; # the highest root
[ 1, 2, 2, 3, 2, 1 ]
```

8 ▶ `GeneratorsOfGroup(` *U* `)` M

This is a special Method for unipotent subgroups of finite Chevalley groups.

9 ▶ `Representative(` *U* `)` M

This method returns an element of the unipotent subgroup U with indeterminates instead of ring elements. Such an element could be used for symbolic computations (see 2.4). The returned element has representation `UNIPOT_DEFAULT_REP` (see 2.3.3).

```
gap> Representative(U_G2);
x_{1}( t_1 ) * x_{2}( t_2 ) * x_{3}( t_3 ) * x_{4}( t_4 ) *
x_{5}( t_5 ) * x_{6}( t_6 )
```

10 ▶ `CentralElement(` *U* `)` M

This method returns the representative of the center of U without calculating the center.

2.3 Elements of unipotent subgroups of Chevalley groups

In this section we will describe the functionality for unipotent elements provided by this package.

1 ► `IsUnipotChevElem(` *elm* `)` C

Category for elements of a unipotent subgroup.

2 ► `IsUnipotChevRepByRootNumbers(` *elm* `)` R
► `IsUnipotChevRepByFundamentalCoeffs(` *elm* `)` R
► `IsUnipotChevRepByRoots(` *elm* `)` R

`IsUnipotChevRepByRootNumbers`, `IsUnipotChevRepByFundamentalCoeffs` and `IsUnipotChevRepByRoots` are different representations for unipotent elements.

Roots of elements with representation `IsUnipotChevRepByRootNumbers` are represented by their numbers (positions) in `PositiveRoots(RootSystem(`*U*`))`.

Roots of elements with representation `IsUnipotChevRepByFundamentalCoeffs` are represented by elements of `PositiveRootsFC(RootSystem(`*U*`))`.

Roots of elements with representation `IsUnipotChevRepByRoots` are represented by roots themself, i.e. elements of `PositiveRoots(RootSystem(`*U*`))`.

(See 2.3.4, 2.3.5 and 2.3.6 for examples.)

3 ► `UNIPOT_DEFAULT_REP` V

This variable contains the default representation for newly created elements, e.g. created by `One` or `Random`. When Unipot is loaded, the default representation is `IsUnipotChevRepByRootNumbers` and can be changed by assigning a new value to `UNIPOT_DEFAULT_REP`.

```
gap> UNIPOT_DEFAULT_REP := IsUnipotChevRepByFundamentalCoeffs;;
```

Note that Unipot doesn't check the type of this value, i.e. you may assign any value to `UNIPOT_DEFAULT_REP`, which may result in errors in following commands:

```
gap> UNIPOT_DEFAULT_REP := 3;;
gap> One( U_G2 );
... Error message ...
```

4 ► `UnipotChevElemByRootNumbers(` *U*`,` *roots*`,` *felems* `)` O
► `UnipotChevElemByRootNumbers(` *U*`,` *root*`,` *felem* `)` O
► `UnipotChevElemByRN(` *U*`,` *roots*`,` *felems* `)` O
► `UnipotChevElemByRN(` *U*`,` *root*`,` *felem* `)` O

`UnipotChevElemByRootNumbers` returns an element of a unipotent subgroup *U* with representation `IsUnipotChevRepByRootNumbers` (see 2.3.2).

roots should be a list of root numbers, i.e. integers from the range 1, ..., `Length(PositiveRoots(RootSystem(`*U*`)))`. And *felems* a list of corresponding ring elements or indeterminates over that ring (see GAP Reference Manual, 64.1.1 for general information on indeterminates or section 2.4 of this manual for examples).

The second variant of `UnipotChevElemByRootNumbers` is an abbreviation for the first one if *roots* and *felems* contain only one element.

`UnipotChevElemByRN` is just a synonym for `UnipotChevElemByRootNumbers`.

```
gap> IsIdenticalObj( UnipotChevElemByRN, UnipotChevElemByRootNumbers );
true
gap> y := UnipotChevElemByRootNumbers(U_G2, [1,5], [2,7] );
x_{1}( 2 ) * x_{5}( 7 )
gap> x := UnipotChevElemByRootNumbers(U_G2, 1, 2);
x_{1}( 2 )
```

In this example we create two elements: $x_{r_1}(2) \cdot x_{r_5}(7)$ and $x_{r_1}(2)$, where $r_i, i = 1, \ldots, 6$ are the positive roots in `PositiveRoots(RootSystem(`*U*`))` and $x_{r_i}(t), i = 1, \ldots, 6$ the corresponding root elements.

5 ► `UnipotChevElemByFundamentalCoeffs(` *U*`,` *roots*`,` *felems* `)` O
► `UnipotChevElemByFundamentalCoeffs(` *U*`,` *root*`,` *felem* `)` O
► `UnipotChevElemByFC(` *U*`,` *roots*`,` *felems* `)` O
► `UnipotChevElemByFC(` *U*`,` *root*`,` *felem* `)` O

`UnipotChevElemByFundamentalCoeffs` returns an element of a unipotent subgroup *U* with representation `IsUnipotChevRepByFundamentalCoeffs` (see 2.3.2).

roots should be a list of elements of `PositiveRootsFC(RootSystem(`*U*`))`. And *felems* a list of corresponding ring elements or indeterminates over that ring (see GAP Reference Manual, 64.1.1 for general information on indeterminates or section 2.4 of this manual for examples).

The second variant of `UnipotChevElemByFundamentalCoeffs` is an abbreviation for the first one if *roots* and *felems* contain only one element.

`UnipotChevElemByFC` is just a synonym for `UnipotChevElemByFundamentalCoeffs`.

```
gap> PositiveRootsFC(RootSystem(U_G2));
[ [ 1, 0 ], [ 0, 1 ], [ 1, 1 ], [ 2, 1 ], [ 3, 1 ], [ 3, 2 ] ]
gap> y1 := UnipotChevElemByFundamentalCoeffs( U_G2, [[ 1, 0 ], [ 3, 1 ]], [2,7] );
x_{[ 1, 0 ]}( 2 ) * x_{[ 3, 1 ]}( 7 )
gap> x1 := UnipotChevElemByFundamentalCoeffs( U_G2, [ 1, 0 ], 2 );
x_{[ 1, 0 ]}( 2 )
```

In this example we create the same two elements as in 2.3.4: $x_{[1,0]}(2) \cdot x_{[3,1]}(7)$ and $x_{[1,0]}(2)$, where $[1,0] = 1r_1 + 0r_2 = r_1$ and $[3,1] = 3r_1 + 1r_2 = r_5$ are the first and the fifth positive roots of `PositiveRootsFC(RootSystem(`*U*`))` respectively.

6 ► `UnipotChevElemByRoots(` *U*`,` *roots*`,` *felems* `)` O
► `UnipotChevElemByRoots(` *U*`,` *root*`,` *felem* `)` O
► `UnipotChevElemByR(` *U*`,` *roots*`,` *felems* `)` O
► `UnipotChevElemByR(` *U*`,` *root*`,` *felem* `)` O

`UnipotChevElemByRoots` returns an element of a unipotent subgroup *U* with representation `IsUnipotChevRepByRoots` (see 2.3.2).

roots should be a list of elements of `PositiveRoots(RootSystem(`*U*`))`. And *felems* a list of corresponding ring elements or indeterminates over that ring (see GAP Reference Manual, 64.1.1 for general information on indeterminates or section 2.4 of this manual for examples).

The second variant of `UnipotChevElemByRoots` is an abbreviation for the first one if *roots* and *felems* contain only one element.

`UnipotChevElemByR` is just a synonym for `UnipotChevElemByRoots`.

```
gap> PositiveRoots(RootSystem(U_G2));
[ [ 2, -1 ], [ -3, 2 ], [ -1, 1 ], [ 1, 0 ], [ 3, -1 ], [ 0, 1 ] ]
gap> y2 := UnipotChevElemByRoots( U_G2, [[ 2, -1 ], [ 3, -1 ]], [2,7] );
x_{[ 2, -1 ]}( 2 ) * x_{[ 3, -1 ]}( 7 )
gap> x2 := UnipotChevElemByRoots( U_G2, [ 2, -1 ], 2 );
x_{[ 2, -1 ]}( 2 )
```

In this example we create again the two elements as in previous examples: $x_{[2,-1]}(2) \cdot x_{[3,-1]}(7)$ and $x_{[2,-1]}(2)$, where $[2, -1] = r_1$ and $[3, -1] = r_5$ are the first and the fifth positive roots of `PositiveRoots(RootSystem(` *U*`))` respectively.

7 ▶ `UnipotChevElemByRootNumbers(` *x* `)` O
▶ `UnipotChevElemByFundamentalCoeffs(` *x* `)` O
▶ `UnipotChevElemByRoots(` *x* `)` O

These three methods are provided for converting a unipotent element to the respective representation.

If x has already the required representation, then x itself is returned. Otherwise a **new** element with the required representation is generated.

```
gap> x;
x_{1}( 2 )
gap> x1 := UnipotChevElemByFundamentalCoeffs( x );
x_{[ 1, 0 ]}( 2 )
gap> IsIdenticalObj(x, x1); x = x1;
false
true
gap> x2 := UnipotChevElemByFundamentalCoeffs( x1 );;
gap> IsIdenticalObj(x1, x2);
true
```

Note: If some attributes of x are known (e.g `Inverse` (see 2.3.15) or `CanonicalForm` (see 2.3.8)), then they are “converted” to the new representation, too.

▶ `UnipotChevElemByRootNumbers(` *U*`,` *list* `) O`
▶ `UnipotChevElemByRoots(` *U*`,` *list* `) O`
▶ `UnipotChevElemByFundamentalCoeffs(` *U*`,` *list* `) O`

DEPRECATED These are old versions of `UnipotChevElemByXX` (from Unipot 1.0 and 1.1). They are deprecated now and exist for compatibility only. They may be removed at any time.

8 ▶ `CanonicalForm(` *x* `)` A

`CanonicalForm` returns the canonical form of x. For more information on the canonical form see Carter [Car89], Theorem 5.3.3 (ii). It says:

Each element of a unipotent subgroup U of a Chevalley group with root system Φ is uniquely expressible in the form

$$\prod_{r_i \in \Phi^+} x_{r_i}(t_i),$$

where the product is taken over all positive roots in increasing order.

```
gap> z := UnipotChevElemByFC( U_G2, [[0,1], [1,0]], [3,2]);
x_{[ 0, 1 ]}( 3 ) * x_{[ 1, 0 ]}( 2 )
gap> CanonicalForm(z);
x_{[ 1, 0 ]}( 2 ) * x_{[ 0, 1 ]}( 3 ) * x_{[ 1, 1 ]}( 6 ) *
x_{[ 2, 1 ]}( 12 ) * x_{[ 3, 1 ]}( 24 ) * x_{[ 3, 2 ]}( -72 )
```

So if we call the positive roots $r_1, \ldots, r_6$, we have $z = x_{r_2}(3)x_{r_1}(2) = x_{r_1}(2)x_{r_2}(3)x_{r_3}(6)x_{r_4}(12)x_{r_5}(24)x_{r_6}(-72)$.

9 ▶ `PrintObj( x )` M
▶ `ViewObj( x )` M

Special methods for unipotent elements. (see GAP Reference Manual, section 6.2 for general information on `View` and `Print`). The output depends on the representation of x.

```
gap> Print(x);
UnipotChevElemByRootNumbers( UnipotChevSubGr( "G", 2, Rationals ), \
[ 1 ], [ 2 ] )gap> View(x);
x_{1}( 2 )gap>

gap> Print(x1);
UnipotChevElemByFundamentalCoeffs( UnipotChevSubGr( "G", 2, Rationals ), \
[ [ 1, 0 ] ], [ 2 ] )gap> View(x1);
x_{[ 1, 0 ]}( 2 )gap>
```

10 ▶ `ShallowCopy( x )` M

This is a special method for unipotent elements.

`ShallowCopy` creates a copy of x. The returned object is **not identical** to x but it is **equal** to x w.r.t. the equality operator `=`. **Note** that `CanonicalForm` and `Inverse` of x (if known) are identical to `CanonicalForm` and `Inverse` of the returned object.

(See GAP Reference Manual, section 12.7 for further information on copyability)

11 ▶ x `=` y M

Special method for unipotent elements. If x and y are identical or are products of the **same** root elements then `true` is returned. Otherwise `CanonicalForm` (see 2.3.8) of both arguments must be computed (if not already known), which may be expensive. If the canonical form of one of the elements must be calculated and `InfoLevel` of `UnipotChevInfo` is at least 3, the user is notified about this:

```
gap> y := UnipotChevElemByRN( U_G2, [1,5], [2,7] );
x_{1}( 2 ) * x_{5}( 7 )
gap> z := UnipotChevElemByRN( U_G2, [5,1], [7,2] );
x_{5}( 7 ) * x_{1}( 2 )
gap> SetInfoLevel( UnipotChevInfo, 3 );
gap> y=z;
#I  CanonicalForm for the 1st argument is not known.
#I                    computing it may take a while.
#I  CanonicalForm for the 2nd argument is not known.
#I                    computing it may take a while.
true
gap> SetInfoLevel( UnipotChevInfo, 1 );
```

12 ▶ x `<` y M

Special Method for `UnipotChevElem`

This is needed e.g. by `AsSSortetList`.

The ordering is computed in the following way: Let $x = x_{r_1}(s_1) \cdots x_{r_n}(s_n)$ and $y = x_{r_1}(t_1) \cdots x_{r_n}(t_n)$, then

$$x < y \quad \Leftrightarrow \quad [s_1, \ldots, s_n] < [t_1, \ldots, t_n],$$

where the lists are compared lexicographically. e.g. for $x = x_{r_1}(1)x_{r_2}(1) = x_{r_1}(1)x_{r_2}(1)x_{r_3}(0)$ (field elems: `[ 1, 1, 0 ]`) and $y = x_{r_1}(1)x_{r_3}(1) = x_{r_1}(1)x_{r_2}(0)x_{r_3}(1)$ (field elems: `[ 1, 0, 1 ]`) we have $y < x$ (above lists ordered lexicographically).

13 ▶ *x* * *y* M

Special method for unipotent elements. The expressions in the form $x_r(t)x_r(u)$ will be reduced to $x_r(t+u)$ whenever possible.

```
gap> y;z;
x_{1}( 2 ) * x_{5}( 7 )
x_{5}( 7 ) * x_{1}( 2 )
gap> y*z;
x_{1}( 2 ) * x_{5}( 14 ) * x_{1}( 2 )
```

Note: The representation of the product will be always the representation of the first argument.

```
gap> x; x1; x=x1;
x_{1}( 2 )
x_{[ 1, 0 ]}( 2 )
true
gap> x * x1;
x_{1}( 4 )
gap> x1 * x;
x_{[ 1, 0 ]}( 4 )
```

14 ▶ `OneOp(` *x* `)` M

Special method for unipotent elements. `OneOp` returns the multiplicative neutral element of x. This is equal to x^0.

15 ▶ `Inverse(` *x* `)` M
▶ `InverseOp(` *x* `)` M

Special methods for unipotent elements. We are using the fact

$$\Big(x_{r_1}(t_1)\cdots x_{r_m}(t_m)\Big)^{-1} = x_{r_m}(-t_m)\cdots x_{r_1}(-t_1).$$

16 ▶ `IsOne(` *x* `)` M

Special method for unipotent elements. Returns `true` if and only if x is equal to the identity element.

17 ▶ *x* ^ *i* M

Integral powers of the unipotent elements are calculated by the default methods installed in GAP. But special (more efficient) methods are instlled for root elements and for the identity.

18 ▶ *x* ^ *y* M

Conjugation of two unipotent elements, i.e. $x^y = y^{-1}xy$. The representation of the result will be the representation of x.

19 ▶ `Comm(` *x*, *y* `)` M
▶ `Comm(` *x*, *y*, `"canonical" )` M

Special methods for unipotent elements.

`Comm` returns the commutator of x and y, i.e. $x^{-1}\cdot y^{-1}\cdot x\cdot y$. The second variant returns the canonical form of the commutator. In some cases it may be more efficient than `CanonicalForm( Comm(` *x*, *y* `) )`

20 ▶ `IsRootElement(` *x* `)` P

`IsRootElement` returns `true` if and only if x is a *root element*, i.e. $x = x_r(t)$ for some root r. We store this property immediately after creating objects.

Note: the canonical form of x may be a root element even if x isn't one.

```
gap> x := UnipotChevElemByRN( U_G2, [1,5,1], [2,7,-2] );
x_{1}( 2 ) * x_{5}( 7 ) * x_{1}( -2 )
gap> IsRootElement(x);
false
gap> CanonicalForm(x); IsRootElement(CanonicalForm(x));
x_{5}( 7 )
true
```

21 ► IsCentral(*U*, *z*)

Special method for a unipotent subgroup and a unipotent element.

2.4 Symbolic computation

In some cases, calculation with explicite elements is not enough. Unipot povides a way to do symbolic calculations with unipotent elements for this purpose. This is done by using indeterminates (see GAP Reference Manual, 64.1 for more information) over the underlying field instead of the field elements.

```
gap> U_G2 := UnipotChevSubGr("G", 2, Rationals);;
gap> a := Indeterminate( Rationals, "a" );
a
gap> b := Indeterminate( Rationals, "b", [a] );
b
gap> c := Indeterminate( Rationals, "c", [a,b] );
c
gap> x := UnipotChevElemByFC(U_G2, [ [3,1], [1,0], [0,1] ], [a,b,c] );
x_{[ 3, 1 ]}( a ) * x_{[ 1, 0 ]}( b ) * x_{[ 0, 1 ]}( c )
gap> CanonicalForm(x);
x_{[ 1, 0 ]}( b ) * x_{[ 0, 1 ]}( c ) * x_{[ 3, 1 ]}( a ) *
x_{[ 3, 2 ]}( a*c )
gap> CanonicalForm(x^-1);
x_{[ 1, 0 ]}( -b ) * x_{[ 0, 1 ]}( -c ) * x_{[ 1, 1 ]}( b*c ) *
x_{[ 2, 1 ]}( -b^2*c ) * x_{[ 3, 1 ]}( -a+b^3*c ) * x_{[ 3, 2 ]}( b^3*c^2 )
```

Bibliography

[Car89] Roger W. Carter. *Simple groups of Lie type*. Wiley Classics Library. John Wiley & Sons Inc., New York, 1989. Reprint of the 1972 original, A Wiley-Interscience Publication.

[GAP04] The GAP Group, Aachen, St Andrews. *GAP – Groups, Algorithms, and Programming, Version 4.4*, 2004.

`http://www.gap-system.org`.

[Hal00] Sergei Haller. Entwicklung und Implementierung eines Algorithmus zur Berechnung von Kommutatoren unipotenter Elemente in Chevalley-Gruppen. Diplomarbeit, Mathematisches Institut, Justus-Liebig-Universität Gießen, April 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”.

\=, 10
*, 10
\<, 10

C

CanonicalForm, 9
CentralElement, 6
Citing Unipot, *3*
Comm, for UnipotChevElem, 11
Conjugation, of UnipotChevElem, 11

E

Elements of unipotent subgroups of Chevalley groups, *6*
Equality, for UnipotChevElem, 10

G

General functionality, *4*
GeneratorsOfGroup, for UnipotChevSubGr, 6

I

Inverse, for UnipotChevElem, 11
InverseOp, for UnipotChevElem, 11
IsCentral, 12
IsOne, 11
IsRootElement, 11
IsUnipotChevElem, 6
IsUnipotChevRepByFundamentalCoeffs, 6
IsUnipotChevRepByRootNumbers, 6
IsUnipotChevRepByRoots, 6
IsUnipotChevSubGr, 4

L

Less than, for UnipotChevElem, 10

M

Multiplication, for UnipotChevElem, 10

N

NegativeRootsFC, 6

O

One, for UnipotChevSubGr, 5
OneOp, for UnipotChevElem, 11
  for UnipotChevSubGr, 5

P

PositiveRootsFC, 6
Powers, of UnipotChevElem, 11
PrintObj, for UnipotChevElem, 9
  for UnipotChevSubGr, 5

R

Representative, 6
RootSystem, for UnipotChevSubGr, 5
Root Systems, *3*

S

ShallowCopy, for UnipotChevElem, 10
Size, for UnipotChevSubGr, 5
Symbolic computation, *12*

U

Unipot, 3
UNIPOT_DEFAULT_REP, 7
UnipotChevElemByFC, 8
UnipotChevElemByFundamentalCoeffs, 8
  element conversion, 9
UnipotChevElemByR, 8
UnipotChevElemByRN, 7
UnipotChevElemByRootNumbers, 7
  element conversion, 9
UnipotChevElemByRoots, 8
  element conversion, 9
UnipotChevInfo, 4
UnipotChevSubGr, 4
Unipotent subgroups of Chevalley groups, *4*

V

ViewObj, for UnipotChevElem, 9
  for UnipotChevSubGr, 5