

Constructing algebraic groups from their Lie algebras

Willem A. de Graaf
Dipartimento di Matematica
Università di Trento
Italy
email: `degraaf@science.unitn.it`

Abstract

A connected algebraic group in characteristic 0 is uniquely determined by its Lie algebra. In this paper an algorithm is given for constructing an algebraic group in characteristic 0, given its Lie algebra. Using this an algorithm is presented for finding a maximal reductive subgroup and the unipotent radical of an algebraic group.

1 Introduction

Due to a wide variety of applications of algebraic groups there is considerable interest in doing computations with such groups. For example, [3], [7] contain both applications of algebraic groups and computational methods. The most natural way to give a linear algebraic group is by a set of polynomial equations. However, it is not easy to determine properties of the group from these equations (for example, whether it is simple, solvable, or nilpotent). A standard way to get around this is to study the Lie algebra of the group. This Lie algebra reflects many of the properties of the group, and can therefore be used to get a picture of the structure of the group. However, by looking at the Lie algebra alone it is for instance not possible to construct subgroups, such as the unipotent radical, or a maximal reductive subgroup. It is the aim of this paper to provide methods for constructing such subgroups by taking the inverse route. In other words, we describe algorithms for going back to the group from its Lie algebra.

For this a few restrictions are necessary. Firstly, in characteristic different from zero it is in general not true that to a given linear Lie algebra corresponds a unique linear algebraic group. There can be more than one. So we restrict to fields of characteristic zero. Second, since the Lie algebra of an algebraic group is equal to the Lie algebra of the connected component of the identity, we restrict to connected algebraic groups. We note that, also with these restrictions, not every Lie algebra corresponds to an algebraic group. Lie algebras that do are called algebraic. There are methods to decide whether or not a given Lie algebra is algebraic (cf. [4]). So we assume that we have given a linear algebraic Lie algebra, and we want to construct the corresponding linear algebraic group.

Now we state the problem more precisely. Let F be a field and let \overline{F} be its algebraic closure. By $R(n, F)$ we denote the polynomial ring $F[x_{11}, \dots, x_{nn}]$ (n^2 indeterminates). A subgroup $G \subset \mathrm{GL}(n, \overline{F})$ is said to be algebraic if there is a set of polynomials $S \subset R(n, \overline{F})$ such that G consists of the $g \in \mathrm{GL}(n, \overline{F})$ with $f(g) = 0$ for all $f \in S$. Let G be an algebraic group, and let $I(G) \subset R(n, \overline{F})$ be the ideal of all polynomials vanishing on G . Then G is said to be defined over F if $I(G)$ is generated by polynomials in $R(n, F)$.

In this paper we take F to be a field of characteristic 0 (\mathbb{Q} for example), and consider algebraic groups that are defined over F . Since F is perfect, any subgroup $G \subset \mathrm{GL}(n, \overline{F})$ defined as the vanishing set of a set of polynomials in $R(n, F)$ is automatically defined over F (cf. [8], §34). We call a set $S \subset R(n, F)$ such that G consists of all g where all elements of S vanish, a set of defining polynomials for G . (We do not assume that they generate $I(G)$.)

The Lie algebra $\mathfrak{g} = \mathrm{Lie}(G)$ of an algebraic group $G \subset \mathrm{GL}(n, \overline{F})$ is a subalgebra of $\mathfrak{gl}(n, \overline{F})$ (the Lie algebra of all $n \times n$ -matrices with coefficients in \overline{F}). Now if G is defined over F , then \mathfrak{g} is a subalgebra of $\mathfrak{gl}(n, F)$. So the problem that we consider is the following. Given an algebraic Lie subalgebra $\mathfrak{g} \subset \mathfrak{gl}(n, F)$, compute a set of defining polynomials (over F) for the unique connected subgroup $G \subset \mathrm{GL}(n, \overline{F})$ such that $\mathfrak{g} = \mathrm{Lie}(G)$.

The algorithm outlined in this paper consists of a few subalgorithms. In Section 2 we describe an algorithm that constructs the smallest algebraic group containing two given algebraic subgroups $G_1, G_2 \subset \mathrm{GL}(n, \overline{F})$. In Section 3 an algorithm is given for constructing the algebraic group corresponding to a Lie algebra that consists of nilpotent matrices. Let $X \in \mathfrak{g}$; then by $G(X)$ we denote the smallest algebraic subgroup of G such that its Lie algebra contains X . In Section 4 we give an algorithm for constructing $G(X)$ in case X is semisimple. Now these algorithms together solve the problem. Indeed, since \mathfrak{g} is algebraic, it is closed under Jordan decomposition. So \mathfrak{g} has a basis consisting of elements that are either nilpotent or semisimple. So for every basis element X we can construct $G(X)$. Let X, Y be two basis elements. Then with the algorithm of Section 2 we can construct the smallest algebraic group H that contains $G(X)$ and $G(Y)$. The Lie algebra of H is generated by X, Y . Then we take a third basis element Z , outside of this subalgebra, and form the smallest algebraic group that contains H and $G(Z)$. Continuing like this eventually we find G .

In Section 5 we give an algorithm to decompose the Lie algebra \mathfrak{g} into a direct sum of a reductive subalgebra, and an ideal consisting of nilpotent elements. Together with the algorithms of the previous sections this yields an algorithm for finding the analogous decomposition of an algebraic group. This is one of the steps needed in the algorithms of [7]. Finally, the last section describes experiences with an implementation of the algorithm in the computer algebra system MAGMA.

2 The algebraic group generated by two subgroups

Let $G_1, G_2 \subset \mathrm{GL}(n, \overline{F})$ be two algebraic groups defined over F . Let $G \subset \mathrm{GL}(n, \overline{F})$ be the smallest algebraic group containing both G_1 and G_2 . Then also G is defined over F (this

follows for example from the algorithm given below). We consider the problem of finding a set of defining polynomials in $R(n, F)$ for G .

First we note that the multiplication map $\cdot : \text{GL}(n, \overline{F}) \times \text{GL}(n, \overline{F}) \rightarrow \text{GL}(n, \overline{F})$ is a morphism of algebraic varieties. Using Gröbner basis techniques we can compute the Zariski closure of the image of this morphism (see, e.g., [6], §1.8.3). This algorithm only uses operations in the base field. Hence the closure of the image of a variety defined over F is defined over F as well. So we can compute defining polynomials in $R(n, F)$ for the Zariski closure $\overline{G_1 G_2}$ of the set $G_1 G_2 = \{g_1 g_2 \mid g_i \in G_i\}$. Now from [2] (Chapter II, §7, Corollary 3), we have the following theorem.

Theorem 1 (Chevalley) *Let G_1, \dots, G_s be connected algebraic subgroups of $\text{GL}(n, \overline{F})$. Then the group H generated by G_1, \dots, G_s is algebraic and connected. Moreover, there exists an $m > 0$ such that every element of H can be written as a product of m elements, each belonging to a G_i .*

This leads to the following algorithm for computing G :

Algorithm 2 *Input: algebraic subgroups $G_1, G_2 \subset \text{GL}(n, \overline{F})$.*

Output: defining polynomials for the smallest algebraic subgroup of $\text{GL}(n, \overline{F})$ containing G_1, G_2 .

1. Let G be the trivial subgroup of $\text{GL}(n, \overline{F})$.
2. Set $G' := G$.
3. Set $G' := \overline{G' G_1}$.
4. Set $G' := \overline{G' G_2}$.
5. If $G' \neq G$ then set $G := G'$ and return to 2. Otherwise return G .

Proof. We show that the algorithm is correct and that it terminates. Let H be the group generated by G_1, G_2 . The identity is always contained in G' . Secondly, if in a round of the iteration G' does not change then $\overline{G' G_1} = G' = \overline{G' G_2}$. We claim that this implies that $G' = H$. It is obvious that $G' \subset H$. Let $g' \in G'$, and $g \in G_1$. Then $g'g \in \overline{G' G_1}$, and consequently $g'g \in G'$. In the same way we see that $g'g \in G'$ if $g \in G_2$. Let $h \in H$. Then by Theorem 1, h can be written as a product of elements from $G_1 \cup G_2$. It follows that $g'h \in G'$ for all $g' \in G'$ and $h \in H$. So since $1 \in G'$ we conclude that $H \subset G'$.

After k rounds of the iteration $G_1 G_2 \cdots G_1 G_2$ (k factors $G_1 G_2$) is a subset of G . So by Theorem 1 there is an $m > 0$ such that after m rounds of the iteration we have $G = H$. But then G' will not change in the next round of the iteration, and the algorithm terminates. \square

3 The nilpotent case

Let $N \subset \mathfrak{gl}(n, F)$ be a matrix Lie algebra consisting of nilpotent matrices. Then N is an algebraic Lie algebra ([1], Chapter V, Proposition 14). By $G(N) \subset \text{GL}(n, \overline{F})$ we denote the connected algebraic group having Lie algebra N . We consider the problem of finding defining polynomials for $G(N)$.

Let N_1, \dots, N_r be a basis of N . Consider the polynomial ring $P = F[T_1, \dots, T_r, x_{ij} \mid 1 \leq i, j \leq n]$. Form the matrix $M = \exp(T_1 N_1 + \dots + T_r N_r)$, and let I be the ideal of P generated by $x_{ij} - M(i, j)$. By elimination techniques using Gröbner bases we can compute the ideal $J = I \cap F[x_{ij}]$.

Proposition 3 *Set $J' = \{f \in F[x_{ij}] \mid f(a) = 0 \text{ for all } a \in G(N)\}$. Then $J = J'$.*

Proof. First of all we note that every element of $G(N)$ can uniquely be written as $\exp(\sum_i \alpha_i N_i)$, where $\alpha_i \in \overline{F}$ ([1], Chapter V, Proposition 14). Now let $f \in J$ and $a = \exp(\sum_i \alpha_i N_i) \in G(N)$. We show that $f(a) = 0$. Consider the ring homomorphism $\varphi : F[T_1, \dots, T_r, x_{ij}] \rightarrow F[x_{ij}]$, defined by sending T_i to α_i , and leaving the x_{ij} alone. Then $\varphi(h) = h$ for all $h \in J$. Also $\varphi(h)(a) = 0$ for all $h \in I$, as this holds for the generators of I . Hence $f(a) = \varphi(f)(a) = 0$. It follows that $J \subset J'$.

Now let $f \in J'$, then $f(\exp(T_1 N_1 + \dots + T_r N_r)) = 0$ because $f(\sum_i \alpha_i N_i) = 0$ where the α_i are arbitrary elements of F . Since $x_{ij} = M(i, j) + (x_{ij} - M(i, j))$, every polynomial in $F[x_{ij}]$ can be written as the sum of a polynomial in T_1, \dots, T_r and an element of I . In particular, $f = p(T_1, \dots, T_r) + h$ for a $h \in I$. Then we substitute $x_{ij} \mapsto M(i, j)$. It follows that $p = 0$ and $f \in I$. Hence $f \in J$. \square

Remark. One of the advantages of this approach is that it returns the vanishing ideal of $G(N)$. In [7] a more direct method is given. However, it does not have the same advantage. It works as follows. Let $p_i \in R(n, F)$ for $1 \leq i \leq m$ be the linear polynomials that define N as a linear subspace of $\mathfrak{gl}(n, F)$. Let $X = (x_{ij})$ be the matrix consisting of the indeterminates x_{ij} . Then $G(N)$ is defined by the polynomial equations

$$(X - 1)^n = 0 \text{ and } p_i(\log^*(X)) \text{ for } 1 \leq i \leq m,$$

where $\log^*(X) = -\sum_{i=1}^{n-1} i^{-1}(1 - X)^i$. However, the resulting polynomials are all of degree $n - 1$, or n . Computing the radical (which is the same as computing the vanishing ideal) of the ideal generated by these polynomials has proved to be extremely time consuming. For larger n it is also rather difficult to construct these polynomials. We constructed the following example. Let \mathfrak{g} be the simple Lie algebra of type A_3 . This Lie algebra has a 10-dimensional irreducible representation. Let x_1, \dots, x_6 denote the matrices of the positive root vectors of \mathfrak{g} in this representation, and let N be the space spanned by them. We tried to construct the above polynomials in MAGMA. But the system crashed when it exceeded the limit of 2GB of memory. The method based on elimination used 136 seconds to construct defining polynomials (see Section 6).

Remark. It is also possible to use Proposition 3 in order to get equations for the groups corresponding to the 1-dimensional Lie algebras spanned by basis elements of N . Then the

algorithm of Section 2 can be used to get equations for $G(N)$. However, experiments show that this leads to a much less efficient algorithm.

4 The semisimple case

Let $X \in \mathfrak{gl}(n, F)$ be a semisimple matrix. By $G(X)$ we denote the smallest algebraic subgroup of $\mathrm{GL}(n, \overline{F})$ such that its Lie algebra contains X . In this section we consider the problem of obtaining defining polynomials for $G(X)$. We will describe an algorithm based on the following theorem.

Theorem 4 (Chevalley) *Let $\alpha_1, \dots, \alpha_n \in \overline{F}$ be the eigenvalues of X . Let $A \in \mathrm{GL}(n, \overline{F})$ be such that $AXA^{-1} = \mathrm{diag}(\alpha_1, \dots, \alpha_n)$. Denote this last matrix by Y . Set*

$$\Lambda = \{(e_1, \dots, e_n) \in \mathbb{Z}^n \mid \sum_{i=1}^n \alpha_i e_i = 0\}.$$

Then

$$G(Y) = \{\mathrm{diag}(c_1, \dots, c_n) \mid c_i \in \overline{F} \text{ and } \prod_i c_i^{e_i} = 1 \text{ for all } (e_1, \dots, e_n) \in \Lambda\}.$$

Furthermore, $G(X) = A^{-1}G(Y)A$.

The first statement is [2], Chapter II, Proposition 2. The second statement is immediate.

The first step in our algorithm will be to construct a finite extension $F' \supset F$ containing the eigenvalues α_i . Then by linear algebra we can construct a basis of the space

$$\Lambda_{\mathbb{Q}} = \{(e_1, \dots, e_n) \in \mathbb{Q}^n \mid \sum_{i=1}^n \alpha_i e_i = 0\}.$$

From this we want to get a basis of Λ . First of all, by multiplying by suitable scalars we get a basis B of $\Lambda_{\mathbb{Q}}$ whose elements have integral coordinates. By B we also denote the $m \times n$ -matrix having the elements of B as rows. Let S be the Smith normal form of B (cf. [12]). This means that we have unimodular matrices P, Q such that $PBQ = S$, with

$$S = \begin{pmatrix} d_1 & & 0 & \cdots & 0 \\ & \ddots & & & \\ & & d_m & 0 & \cdots & 0 \end{pmatrix},$$

where the d_i are positive integers.

Lemma 5 *B is a basis of Λ if and only if all d_i are equal to 1.*

Proof. Suppose that all $d_i = 1$. Let $\Lambda' \subset \mathbb{Z}^n$ be the subgroup generated by B . Then there is a surjective homomorphism $f : \mathbb{Z}^n \rightarrow \mathbb{Z}^{n-m}$ with kernel Λ' ([12], Proposition 3.3 of Chapter 8). Let $v \in \Lambda$, and write v as a linear combination of elements of B with rational coefficients. After multiplying by a suitable integer s we see that $sv \in \Lambda'$. But then $0 = f(sv) = sf(v)$. Hence $f(v) = 0$ and $v \in \Lambda'$. We conclude that $\Lambda = \Lambda'$.

Now suppose that B is a basis of Λ . We have a surjective homomorphism $f : \mathbb{Z}^n \rightarrow \mathbb{Z}/d_1\mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_m\mathbb{Z} \oplus \mathbb{Z}^{n-m}$, with kernel Λ . Suppose that i is such that $d_i > 1$. Then there is a $v \in \mathbb{Z}^n$ such that $f(v) \neq 0$ but $f(d_iv) = 0$. In other words, $d_iv \in \Lambda$. But then $v \in \Lambda$, and we have a contradiction. \square

Note that $B = P^{-1}SQ^{-1}$. This means that d_i times the i -th row of Q^{-1} lies in the span of B , and hence in Λ . Therefore, the i -th row of Q^{-1} itself lies in Λ , for $1 \leq i \leq m$. Set

$$S' = \begin{pmatrix} 1 & & 0 & \cdots & 0 \\ & \ddots & & & \\ & & 1 & 0 & \cdots & 0 \end{pmatrix},$$

and $A = P^{-1}S'Q^{-1}$. Then the rows of A belong to Λ , and form a basis of $\Lambda_{\mathbb{Q}}$. Moreover, the Smith normal form of A is S' . Hence by Lemma 5, the rows of A form a basis of Λ . We remark that there are efficient algorithms for computing the matrices P, Q (see [12]). Hence we can compute a basis of Λ , given a basis of $\Lambda_{\mathbb{Q}}$.

Let $A_I(X)$ be the associative algebra with one generated by X . Then $A_I(X)$ is spanned by I, X, X^2, \dots, X^t , where $t+1$ is the degree of the minimal polynomial of X . By [2], §13, Theorem 10, $G(X) \subset A_I(X)$. Let α_i, Y be as in Theorem 4. We note that the minimal polynomial of a semisimple matrix is the square free part of its characteristic polynomial. In particular, it does not depend on the base field, but only on the coefficients of the matrix. So $A_I(Y)$ is spanned by I, Y, \dots, Y^t .

Let $y = \sum_{i=0}^t \delta_i Y^i \in A_I(Y)$. Then by $y(k, k)$ we denote the entry on position (k, k) . By Theorem 4 we have that $y \in G(Y)$ if and only if $\prod_k y(k, k)^{e_k} = 1$ for (e_1, \dots, e_n) in a basis of Λ . Now set $e'_k = e_k$ if $e_k \geq 0$, and $e'_k = 0$ otherwise. Also $e''_k = e'_k - e_k$. Then $\prod_k y(k, k)^{e_k} = 1$ if and only if $\prod_k y(k, k)^{e'_k} = \prod_k y(k, k)^{e''_k}$. In this we substitute $y(k, k) = \sum_i \delta_i \alpha_k^i$. This yields a polynomial equation for the δ_i with coefficients in F' . By writing the coefficients as linear combinations of a basis of F' over F , we get polynomials p_1, \dots, p_s , $p_k \in F[T_0, \dots, T_t]$, with the property that $\prod_k y(k, k)^{e_k} = 1$ if and only if $p_i(\delta_0, \dots, \delta_t) = 0$ for $1 \leq i \leq s$. Let h_1, \dots, h_m be the totality of these polynomials that we get when we let (e_1, \dots, e_n) run through a basis of Λ . Then $y \in G(Y)$ if and only if $h_i(\delta_0, \dots, \delta_t) = 0$ for $1 \leq i \leq m$.

Note that $y \in G(Y)$ if and only if $A^{-1}yA \in G(X)$ (where A is as in Theorem 4). But $A^{-1}yA = \sum_i \delta_i X^i$. We conclude that $\sum_i \delta_i X^i \in G(X)$ if and only if $h_i(\delta_0, \dots, \delta_t) = 0$ for $1 \leq i \leq m$.

We summarise these findings in the following algorithm:

Algorithm 6 *Input: a semisimple matrix $X \in \mathfrak{gl}(n, F)$.
Output: defining polynomials for $G(X)$.*

1. Construct a finite extension $F' \supset F$ containing the eigenvalues of X .
2. Compute a basis of Λ .
3. Construct polynomials $h_1, \dots, h_m \in F[T_0, \dots, T_t]$ with the property that $\sum_i \delta_i X^i \in G(X)$ if and only if $h_i(\delta_0, \dots, \delta_t) = 0$ for $1 \leq i \leq m$.
4. Set $M = \sum_{i=0}^t T_i X^i$ and let $\alpha_{ij}^k \in F$ be such that $T_k = \sum_{ij} \alpha_{ij}^k M(i, j)$.
5. Let x_{ij} for $1 \leq i, j \leq n$ be indeterminates, and consider the substitution $T_k \mapsto \sum_{ij} \alpha_{ij}^k x_{ij}$. Let $\varphi : F[T_0, \dots, T_t] \rightarrow F[x_{ij}]$ be the corresponding ring homomorphism.
6. Let $g_1, \dots, g_r \in F[x_{ij}]$ be linear polynomials with the property that $a \in \mathfrak{gl}(n, F)$ lies in $A_I(X)$ if and only if $g_i(a) = 0$ for $1 \leq i \leq r$.
7. Return $\{g_1, \dots, g_r\} \cup \{\varphi(h_1), \dots, \varphi(h_m)\}$.

Proof. Steps 1, 2, and 3 have already been commented on. Since the X^i for $0 \leq i \leq t$ are linearly independent, the coefficients α_{ij}^k in Step 4 exist (but they are not necessarily unique). Let $a = (a_{ij}) \in \text{GL}(n, F)$ be such that $g_i(a) = 0$ for all i . Then $a \in A_I(X)$ so $a = \sum_i \delta_i X^i$. By substituting $T_k \mapsto \delta_k$ in the equation $T_k = \sum_{ij} \alpha_{ij}^k M(i, j)$ we see that $\delta_k = \sum_{i,j} \alpha_{ij}^k a_{ij}$. It follows that $a \in G(X)$ if and only if

$$h_r\left(\sum_{ij} \alpha_{ij}^0 a_{ij}, \dots, \sum_{ij} \alpha_{ij}^t a_{ij}\right) = 0$$

for $1 \leq r \leq m$. But this is equivalent to $\varphi(h_r)(a) = 0$ for $1 \leq r \leq m$. We conclude that this algorithm returns defining polynomials for $G(X)$. \square

Example 7 Let

$$X = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

Then $A_I(X)$ is spanned by I, X . The eigenvalues of X are $i, -i$, and hence Λ is spanned by $(1, 1)$. Let $Y = \text{diag}(i, -i)$ and $y = \delta_0 I + \delta_1 Y$, then $y(1, 1) = \delta_0 + \delta_1 i$, $y(2, 2) = \delta_0 - \delta_1 i$. Since $\dim \Lambda = 1$, there is only one equation that needs to be satisfied, i.e., $y(1, 1)y(2, 2) = 1$, or $\delta_0^2 + \delta_1^2 = 1$. Now

$$M = T_0 I + T_1 X = \begin{pmatrix} T_0 & T_1 \\ -T_1 & T_0 \end{pmatrix}.$$

So we get the substitution $T_0 = x_{11}$ and $T_1 = x_{12}$. The equation above then transforms to $x_{11}^2 + x_{12}^2 - 1 = 0$. Furthermore, the linear equations of Step 6 of the algorithm are $x_{21} + x_{12} = 0$, $x_{11} - x_{22} = 0$.

5 An application

Let $G \subset \mathrm{GL}(n, \overline{F})$ be an algebraic group, defined over F . Then there is a unique maximal unipotent subgroup, $u(G)$, called the unipotent radical of G . Furthermore, there is a reductive subgroup $H \subset G$ with $G = H \ltimes u(G)$. In [7] algorithms are given for finding defining polynomials for H and $u(G)$. However, these algorithms do not seem to be very practical. In this section we describe a method based on the algorithms of the previous sections.

Set $\mathfrak{g} = \mathrm{Lie}(G)$. Let \mathfrak{s} denote the solvable radical of \mathfrak{g} , and \mathfrak{l} a Levi subalgebra of \mathfrak{g} , and \mathfrak{n} the largest ideal of \mathfrak{g} consisting of nilpotent elements. Then by [1], Chapter V, §4, Proposition 5 (see also [11]), \mathfrak{s} has a commutative subalgebra \mathfrak{d} consisting of semisimple elements, with the following properties

1. $\mathfrak{s} = \mathfrak{d} + \mathfrak{n}$ (semidirect sum),
2. $[\mathfrak{l}, \mathfrak{d}] = 0$,
3. \mathfrak{n} is the set consisting of all nilpotent elements of \mathfrak{s} .

Now H and $u(G)$ are the algebraic groups corresponding to $\mathfrak{l} + \mathfrak{d}$ and \mathfrak{n} respectively. There are algorithms to compute \mathfrak{l} and \mathfrak{s} (cf. [5]). In the remainder of this section we describe how to compute \mathfrak{d} and \mathfrak{n} . Then by using the algorithms in the first part of this paper, we can compute defining polynomials for H and $u(G)$.

Lemma 8 *Set $\mathfrak{h} = C_{\mathfrak{s}}(\mathfrak{d})$, the centralizer of \mathfrak{d} in \mathfrak{s} . Then \mathfrak{h} is a Cartan subalgebra of \mathfrak{s} , and $\mathfrak{h} = \mathfrak{d} + C_{\mathfrak{n}}(\mathfrak{d})$.*

Proof. Let $x \in C_{\mathfrak{s}}(\mathfrak{d})$, then we can write $x = y + z$, where $y \in \mathfrak{d}$ and $z \in \mathfrak{n}$. So for any $u \in \mathfrak{d}$ we get $0 = [u, x] = [u, y] + [u, z] = [u, z]$. It follows that $z \in C_{\mathfrak{s}}(\mathfrak{d})$. Hence $\mathfrak{h} = \mathfrak{d} + C_{\mathfrak{n}}(\mathfrak{d})$. Since \mathfrak{n} is nilpotent and $[\mathfrak{d}, C_{\mathfrak{n}}(\mathfrak{d})] = 0$ it also follows that \mathfrak{h} is nilpotent.

Now let $x \in \mathfrak{d}$. Then x is a semisimple linear transformation. This implies that also $\mathrm{ad}x : \mathfrak{s} \rightarrow \mathfrak{s}$ is semisimple. Hence $C_{\mathfrak{s}}(\mathfrak{d}) = \mathfrak{s}_0(\mathrm{ad}\mathfrak{d})$, where

$$\mathfrak{s}_0(\mathrm{ad}\mathfrak{d}) = \{y \in \mathfrak{s} \mid (\mathrm{ad}x)^{\dim \mathfrak{s}}(y) = 0 \text{ for all } x \in \mathfrak{d}\}.$$

By [13], Theorem 4.4.4.8 every Cartan subalgebra of $\mathfrak{s}_0(\mathrm{ad}\mathfrak{d})$ is a Cartan subalgebra of \mathfrak{s} . But as seen above $\mathfrak{h} = \mathfrak{s}_0(\mathrm{ad}\mathfrak{d})$ is nilpotent. So it is its own Cartan subalgebra. \square

Lemma 9 *Let \mathfrak{h}' be a Cartan subalgebra of \mathfrak{s} . Then there is an abelian subalgebra $\mathfrak{d}' \subset \mathfrak{s}$ consisting of semisimple elements, such that $\mathfrak{s} = \mathfrak{d}' + \mathfrak{n}$ (semidirect sum), and $\mathfrak{h}' = C_{\mathfrak{s}}(\mathfrak{d}') = \mathfrak{d}' + C_{\mathfrak{n}}(\mathfrak{d}')$.*

Proof. Let D be the group of automorphisms of \mathfrak{s} generated by $\exp(\mathrm{ad}x)$ for $x \in [\mathfrak{s}, \mathfrak{s}]$. Set $\mathfrak{h} = C_{\mathfrak{s}}(\mathfrak{d})$, which is a Cartan subalgebra of \mathfrak{s} by Lemma 8. By [1], Chapter VI, Proposition 19, there is a $\sigma \in D$ such that $\sigma(\mathfrak{h}) = \mathfrak{h}'$. Set $\mathfrak{d}' = \sigma(\mathfrak{d})$. In order to prove that \mathfrak{d}' consists of semisimple elements we may assume that $\sigma = \exp(\mathrm{ad}x)$, for an $x \in [\mathfrak{s}, \mathfrak{s}]$. Then for

$u \in \mathfrak{d}$ we have $\sigma(u) = (\exp x)u(\exp x)^{-1}$ (by [9], Chapter IX, (38); note that $[\mathfrak{s}, \mathfrak{s}] \subset \mathfrak{n}$ so $\exp(x)$ is a well defined endomorphism). So \mathfrak{d}' is obtained from \mathfrak{d} by conjugation with a fixed endomorphism, and hence consists of semisimple elements. The other properties follow from the analogous properties of \mathfrak{h} and the fact that σ is an automorphism. \square

Let \mathfrak{r} be a finite-dimensional Lie algebra, and let $\mathfrak{a} \subset \mathfrak{r}$ be a nilpotent subalgebra. Then \mathfrak{r} has a Fitting decomposition with respect to the adjoint action of \mathfrak{a} (cf. [9], [5]). This decomposition is written $\mathfrak{r} = \mathfrak{r}_0(\mathfrak{a}) \oplus \mathfrak{r}_1(\mathfrak{a})$. These are called the Fitting 0-component and Fitting 1-component respectively. They are defined as $\mathfrak{r}_0(\mathfrak{a}) = \{x \in \mathfrak{r} \mid \text{for all } y \in \mathfrak{a} \text{ there is an } i > 0 \text{ such that } (\text{ad}(y))^i(x) = 0\}$, and $\mathfrak{r}_1(\mathfrak{a}) = \bigcap_{i>0} [\mathfrak{a}^i, \mathfrak{r}]$, where by $[\mathfrak{a}^i, \mathfrak{r}]$ we denote the space $[\mathfrak{a}, [\mathfrak{a}, \dots, [\mathfrak{a}, \mathfrak{r}] \dots]]$ (i factors \mathfrak{a}). Based on this there is a straightforward algorithm for computing a basis of $\mathfrak{r}_1(\mathfrak{a})$ (see [5]).

Lemma 10 *Let \mathfrak{h}' and \mathfrak{d}' be as in the previous lemma. Let a_1, \dots, a_r be a basis of \mathfrak{h}' and let $a_i = s_i + n_i$ be the Jordan decomposition of a_i . Then \mathfrak{d}' is spanned by s_1, \dots, s_r . Let $\mathfrak{s}_1(\mathfrak{h}')$ be the Fitting 1-component of \mathfrak{s} with respect to the adjoint action of \mathfrak{h}' . Then \mathfrak{n} is spanned by the n_i along with $\mathfrak{s}_1(\mathfrak{h}')$.*

Proof. Let $h \in \mathfrak{h}'$; then by Lemma 9, there are $s \in \mathfrak{d}'$ and $n \in C_{\mathfrak{n}}(\mathfrak{d}')$ with $h = s + n$. But then s is semisimple, n is nilpotent and $[s, n] = 0$. It follows that $h = s + n$ is the Jordan decomposition of h . In particular, all s_i lie in \mathfrak{d}' and all n_i lie in $C_{\mathfrak{n}}(\mathfrak{d}')$. Also $h = \sum_i \alpha_i a_i = (\sum_i \alpha_i s_i) + (\sum_i \alpha_i n_i)$. This implies that $\sum_i \alpha_i s_i = s$ and $\sum_i \alpha_i n_i = n$. Therefore, the s_i span \mathfrak{d}' , and the n_i span $C_{\mathfrak{n}}(\mathfrak{d}')$. Note that $\mathfrak{s}_0(\mathfrak{h}') = \mathfrak{h}'$ as \mathfrak{h}' is a Cartan subalgebra of \mathfrak{s} (cf. [9], Chapter III, Proposition 1). Also $\mathfrak{s}_1(\mathfrak{h}') \subset [\mathfrak{s}, \mathfrak{s}] \subset \mathfrak{n}$. Now $\mathfrak{s} = \mathfrak{d}' \oplus C_{\mathfrak{n}}(\mathfrak{d}') \oplus \mathfrak{s}_1(\mathfrak{h}')$. The last two spaces are contained in \mathfrak{n} . Since we also have $\mathfrak{s} = \mathfrak{d}' + \mathfrak{n}$ we get that $\mathfrak{n} = C_{\mathfrak{n}}(\mathfrak{d}') \oplus \mathfrak{s}_1(\mathfrak{h}')$, whence the last statement. \square

We note that there are algorithms for computing a Cartan subalgebra of a Lie algebra of characteristic 0 (cf. [5]). So the previous lemma yields a straightforward algorithm for computing bases of \mathfrak{d}' and \mathfrak{n} .

6 Practical experiences

In this section we report on practical experiences with an implementation of the algorithms in the computer algebra system MAGMA. The computations were done on a 2GHz processor, with 2GB of memory.

First we consider the algorithm from Section 3. This algorithm takes as input a set of nilpotent matrices that span a Lie algebra, and returns defining polynomials for the corresponding algebraic group. In order to generate input we have constructed irreducible representations of several simple Lie algebras, and taken the matrices of the positive root vectors. The running times of the algorithm are listed in Table 1. From the table we see that the algorithm performs quite well for small matrices. However, the running times increase sharply when the size of the matrices increases. This is due to the Gröbner basis calculation, used for the elimination of variables.

type	dim rep.	dim N	time $G(N)$
B_2	5	4	0.08
B_2	10	4	43
G_2	7	6	3.45
A_3	10	6	136
C_3	14	9	∞
D_4	8	12	∞

Table 1: Running times for the algorithm of Section 3. The first column lists the type of the simple Lie algebra, the second column has the dimension of the irreducible module, the third column the dimension of the Lie algebra spanned by the positive root vectors. The last column displays the time (in seconds) for constructing defining polynomials for the group. An ∞ means that the algorithm did not complete with 2GB of memory.

polynomial	dimension splitting field	time $G(X)$
$x^8 - x^7 + x^6 + 2x^5 - 3x^4 + 4x^3 + 2$	360	83
$x^6 - 2x^4 + x^2 - 2x - 1$	360	293
$x^8 + 12x^6 + 50x^4 + 83x^2 + 43$	384	89
$x^8 - 2x^6 + 7x^4 - 8x^2 - 4x + 7$	576	1463

Table 2: Running times for the algorithm of Section 4. The first column lists the polynomial, and the second column the degree of its splitting field. The last column displays the time (in seconds) for constructing defining polynomials for the group corresponding to the companion matrix of the polynomial.

Next we consider the algorithm of Section 4. Here the input is a single semisimple matrix. As input we have used the companion matrices of several square free polynomials (except the first one these are taken from the database in [10]). The running times are displayed in Table 2. As was to be expected the algorithm spends a lot more time if the degree of the splitting field gets larger.

Finally we consider the algorithm of Section 2. For this algorithm we have constructed several rather different inputs, which we describe separately.

Let A be an algebra, and $\text{Der}(A)$ the Lie algebra of its derivations. By [2], Chapter II, §14, Theorem 16, we know that $\text{Der}(A)$ is the Lie algebra of the automorphism group of A . Here we consider the algebra A of quaternions over \mathbb{Q} . This algebra has basis $1, i, j, k$, where 1 is the identity. The other product relations are $ij = k, ji = -k, ik = -j, ki = j, jk = i, kj = -i$. Then $\text{Der}(A)$ is spanned by

$$x_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, x_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix}, x_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix},$$

(where we use the column convention to represent a linear map with respect to the basis $1, i, j, k$ of A). Let H_i be the smallest algebraic group containing x_i . The x_i are semisimple, so equations for the H_i can be computed using the algorithm of Section 4. Let G be the smallest algebraic subgroup of $\text{GL}(4, \mathbb{C})$ containing H_1, H_2 . The Lie algebra of G contains x_1, x_2 . But then it contains also x_3 . It follows that its Lie algebra is equal to $\text{Der}(A)$. Hence G is the connected component of the identity of the automorphism group of A . The computation of equations for G (given those of H_1 and H_2) costs 8.3 seconds.

The next example has the Lie algebra consisting of the matrices

$$\begin{pmatrix} * & * & * & * \\ * & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{pmatrix}.$$

Let H_1 denote the group corresponding to the subalgebra consisting of strictly upper triangular matrices. By $H_{i,j}$ we denote the group corresponding to the subalgebra spanned by $e_{i,j}$ (where $e_{i,j}$ denotes the matrix with 1 on position (i, j) and zeros elsewhere). Let G_1 denote the smallest algebraic group containing both H_1 and $H_{2,1}$. The construction of defining polynomials for G_1 costs 0.99 seconds. Let G_2 denote the smallest algebraic group containing G_1 and $H_{4,3}$. The computation of defining polynomials for G_2 costs 2.4 seconds. Let G_3 be the smallest algebraic group containing G_2 and $H_{1,1}$. The computation of defining polynomials for G_3 costs 5.4 seconds. Let $G = G_4$ denote the smallest algebraic group containing G_3 and $H_{3,3}$. The computation of defining polynomials for G_4 costs 15 seconds. We note that G is equal to the algebraic group corresponding to the whole Lie algebra.

Let L be the Lie algebra of type B_2 in its 4-dimensional representation. Let H_1 and H_2 denote the groups corresponding to the subalgebra spanned by the positive and negative root vectors respectively. Let G be the smallest algebraic group containing both H_1 and H_2 . The computation of defining polynomials for G takes 173 seconds.

Finally, let L be the simple Lie algebra of type A_3 in its 4-dimensional representation. Let H_1, H_2 and G be as in the previous example. The algorithm did not manage to compute defining polynomials for G within 2GB.

Concluding we can say that the algorithms do work for small examples. However, due to the Gröbner basis computations, needing a lot of memory and running time, the algorithms are not yet practical for larger examples.

References

- [1] C. Chevalley. *Théorie des Groupes de Lie, Tome III. Théorèmes généraux sur les algèbres de Lie*. Hermann, Paris, 1955.
- [2] Claude Chevalley. *Théorie des groupes de Lie. Tome II. Groupes algébriques*. Actualités Sci. Ind. no. 1152. Hermann & Cie., Paris, 1951.

- [3] Harm Derksen, Emmanuel Jeandel, and Pascal Koiran. Quantum automata and algebraic groups. *J. Symbolic Comput.*, 39(3-4):357–371, 2005.
- [4] Claus Fieker and Willem de Graaf. Constructing algebraic Lie algebras. preprint, 2006, <http://arxiv.org/abs/math.RA/0611414>.
- [5] W. A. de Graaf. *Lie Algebras: Theory and Algorithms*, volume 56 of *North-Holland Mathematical Library*. Elsevier Science, 2000.
- [6] Gert-Martin Greuel and Gerhard Pfister. *A Singular introduction to commutative algebra*. Springer-Verlag, Berlin, 2002. With contributions by Olaf Bachmann, Christoph Lossen and Hans Schönemann, With 1 CD-ROM (Windows, Macintosh, and UNIX).
- [7] Fritz Grunewald and Daniel Segal. Some general algorithms. I. Arithmetic groups. *Ann. of Math. (2)*, 112(3):531–583, 1980.
- [8] J. E. Humphreys. *Linear Algebraic Groups*. Springer Verlag, New York, Heidelberg, Berlin, 1975.
- [9] N. Jacobson. *Lie Algebras*. Dover, New York, 1979.
- [10] Jürgen Klüners and Gunter Malle. A database for field extensions of the rationals. *LMS J. Comput. Math.*, 4:182–196 (electronic), 2001.
- [11] G. D. Mostow. Fully reducible subgroups of algebraic groups. *Amer. J. Math.*, 78:200–221, 1956.
- [12] C. C. Sims. *Computation with Finitely Presented Groups*. Cambridge University Press, Cambridge, 1994.
- [13] D. J. Winter. *Abstract Lie Algebras*. M.I.T. Press, Cambridge, Mass., 1972.