COMPUTING IN GROUPS OF LIE TYPE

ARJEH M. COHEN & SCOTT H. MURRAY & D. E. TAYLOR

Abstract. We describe two methods for computing with the elements of untwisted groups of Lie type: using the Steinberg presentation and highest weight representations. We give algorithms for element arithmetic within the Steinberg presentation. Conversion between this presentation and linear representations is achieved using a new generalisation of row and column reduction.

1. Introduction

The groups of Lie type are among the most important structures in modern mathematics. Examples of such groups include reductive Lie groups, reductive algebraic groups, and finite groups of Lie type (which include most of the finite simple groups). Many problems in the representation theory of groups of Lie type have been solved using computers (van Leeuwen et al., 1992; Geck et al., 1996). In this paper, we give methods for computing within the groups themselves, a problem which has only recently been tackled (Haller, 2000; Riebeek, 1998). We have implemented these algorithms in the Magma computer algebra system (Bosma and Cannon, 1997). We confine our attention to the untwisted groups; automorphisms and twisted groups will be dealt with in subsequent work.

Computational group theory has three main strands: permutation groups, matrix groups, and finitely presented groups. The permutation group approach has traditionally been the most effective because of the very efficient Schreier-Sims family of algorithms (Butler, 1991). Recently, effective algorithms for matrix groups over finite fields have started to appear, in particular, the matrix group recognition project based on Aschbacher's theorem (Leedham-Green, 2001). The Todd-Coxeter and Knuth-Bendix algorithms for finitely presented groups (Sims, 1994) are the oldest methods in computational group theory.

Permutation group methods are clearly useless for infinite groups of Lie type. Even for finite groups of Lie type, these methods soon become impractical; for example, the smallest permutation representation of $E_8(2)$ has degree

$$293,091,386,578,365,375 \approx 2.9 \times 10^{17},$$

well beyond the reach of existing computers (Liebeck and Saxl, 1987).

On the other hand, for any field $\mathbb{F}$, there is a faithful linear representation of $E_8(\mathbb{F})$ over $\mathbb{F}$ of degree 248, which makes it feasible to do element operations (provided we can compute in $\mathbb{F}$). Howlett et al. (1999) construct matrix representations for finite Chevalley groups; this has been implemented in Magma.

This paper was written during a stay of the first two authors at the University of Sydney. They wish to thank the institute for its hospitality.
MSC classifications: 20G15, 20C40.

The groundbreaking work of Steinberg (1962, 1968) gives a presentation for the groups of Lie type. In the presentation for $E_8(\mathbb{F})$, each element can be expressed as a word of length at most 368. This does not necessarily give us a finitely presented group, however, because some of our generators are parametrised by the field (more precisely, 240 terms are parametrised by the additive group $\mathbb{F}$, 8 are parametrised by the multiplicative group $\mathbb{F}^\times$, and 120 are field independent). A word in this presentation requires less memory than a matrix representation, except for type A_n where the memory usage is asymptotically the same. An additional advantage is that there is a normal form for elements (the Bruhat decomposition) which reflects the Lie theoretic structure of the group, thus facilitating the use of Lie theoretic techniques. Another computationally important presentation for finite groups of Lie type is described in (Babai et al., 1997; Hulpke and Seress, 2001).

In this paper, we describe methods for doing arithmetic in the Steinberg presentation over any field $\mathbb{F}$ in which computation is possible. We note that Riebeek (1998), mainly for $\mathbb{F} = \mathbb{F}_2$, and Haller (2000), mainly for unipotent subgroups, have done part of this work before. We also give algorithms for conversion between the Steinberg presentation and highest weight representations over the field $\mathbb{F}$. Converting a word to a matrix builds on the methods of de Graaf (2001) for Lie algebras. Converting a matrix to a word requires a generalisation of row and column reduction—this is the main result of this paper. Given these conversion algorithms, we can compute with either the presentation or a linear representation, whichever is most effective for the problem at hand.

A motivating application for this work is the matrix group recognition project (Leedham-Green, 2001), which provides a framework for decomposing a matrix group over a finite field into a subnormal subgroup chain whose quotients are almost simple. Our methods are useful for this project since most almost simple groups are finite groups of Lie type. Firstly the Steinberg presentation gives a computer representation of these groups in which conjugacy classes and other such properties can be computed efficiently using Lie theory. Secondly our generalised row and column reduction algorithm gives a membership test for these groups as matrix groups, which is necessary for recognising simple groups of Lie type (Kantor and Seress, 2001).

We use standard group theoretic notation (Alperin and Bell, 1995). Our group actions are on the right, so conjugation is given by $x^y = y^{-1}xy$ and the commutator by $[x, y] = x^{-1}y^{-1}xy$. We occasionally use left conjugation: ${}^y x = yxy^{-1}$.

In Sections 2 and 3, we discuss the computation of root systems and structure constants. Section 4 describes the Steinberg presentation. Section 5 contains element arithmetic and canonical form algorithms for groups of Lie type; an implementation of these algorithms is available in release 2.8 of Magma. We describe algorithms for converting between the Steinberg presentation and highest weight representations in Sections 6 and 7; these will be available in the next release of Magma.

2. Root data

A root system is the basic structure used for the classification of complex semisimple Lie algebras; a pair of root systems in duality is the structure used for the classification of reductive algebraic groups—such a pair is called a root datum. We

assume that all permutation and matrix actions are on the right. This is in keeping with the conventions of computational group theory but leads to differences between our formulas and those found in standard references such as Demazure (1965), Carter (1993), and Springer (1998).

2.1. **Definition.** Let X and Y be free $\mathbb{Z}$-modules of rank d with a bilinear pairing $\langle \circ, \circ \rangle : X \times Y \to \mathbb{Z}$ putting them in duality. Assume we have a basis $e_1, \ldots, e_d$ for X and a dual basis $f_1, \ldots, f_d$ for Y, so that $\langle e_i, f_j \rangle = \delta_{ij}$. Let Φ be a finite subset of X and suppose that for each α in Φ we have a corresponding $\alpha^\star$ in Y; set $\Phi^\star = \{\alpha^\star \mid \alpha \in \Phi\}$ and define $\alpha^{\star\star} = \alpha$. We call the elements of Φ *roots* and the elements of $\Phi^\star$ *coroots*.

Given a root α we define linear maps $s_\alpha : X \to X$ by $x s_\alpha = x - \langle x, \alpha^\star \rangle \alpha$ and $s_\alpha^\star : Y \to Y$ by $y s_\alpha^\star = y - \langle \alpha, y \rangle \alpha^\star$. These maps are *reflections* if one of the following equivalent properties hold: $\langle \alpha, \alpha^\star \rangle = 2$; $s_\alpha^2 = 1$; $\langle x s_\alpha, y s_\alpha^\star \rangle = \langle x, y \rangle$ for all $x \in X$ and $y \in Y$; $\alpha s_\alpha = -\alpha$.

We say that $\mathcal{R} = (X, \Phi, Y, \Phi^\star)$ is a *root datum* if the following are satisfied for every root α in Φ: s_α and $s_\alpha^\star$ are reflections; Φ is closed under the action of s_α; and $\Phi^\star$ is closed under the action of $s_\alpha^\star$. Furthermore, we require that all our root data be *reduced*, ie. if α and β are roots with β a multiple of α, then $\beta = \pm\alpha$. The *Weyl group* W is the group generated by the reflections s_α.

2.2. **Simple roots and positive roots.** The roots $\alpha_1, \ldots, \alpha_n$ are *simple roots* for $\mathcal{R}$ if they are a basis of $\mathbb{Q}\Phi \leq \mathbb{Q} \otimes X$ and $\Phi = \Phi^+ \cup \Phi^-$, where Φ^+ is the set of roots that are linear combinations of the simple roots with nonnegative coefficients and $\Phi^- = -\Phi^+$. The elements of Φ^+ are called *positive roots* and the elements of Φ^- are called *negative roots*. The coroots corresponding to the simple (resp. positive, negative) roots are called simple (resp. positive, negative) coroots.

We call n the *rank* and d the *dimension* of $\mathcal{R}$; if $n = d$ then $\mathcal{R}$ is *semisimple*. The *Cartan matrix* of $\mathcal{R}$ is $C = [\langle \alpha_i, \alpha_j^\star \rangle]_{i,j=1}^n$. Given a root $\alpha = \sum_{i=1}^n a_i \alpha_i$, we define its *height* by $h(\alpha) = \sum_{i=1}^n a_i$. The *fundamental weights* $\omega_1, \ldots, \omega_n$ are the elements of $\mathbb{Q}\Phi$ such that $\langle \omega_i, \alpha_j^\star \rangle = \delta_{ij}$. An element λ of $\mathbb{Q} \otimes X$ is *dominant* if $\langle \lambda, \alpha_i^\star \rangle \geq 0$ for all i.

Given an element w in the Weyl group W, define

$$\Phi_w = \{\alpha \in \Phi^+ \mid \alpha w^{-1} \in \Phi^-\} = \Phi^+ \cap \Phi^- w.$$

The *length* $l(w)$ is the cardinality of Φ_w; W contains a unique longest element, which we denote w_0.

Lemma 2.1.

(1) Given a reduced expression $w = s_{\beta_1} \cdots s_{\beta_l}$ we have

$$\Phi_w = \{\ \beta_l,\ \beta_{l-1} s_{\beta_l},\ \beta_{l-2} s_{\beta_{l-1}} s_{\beta_l},\ \ldots,\ \beta_1 s_{\beta_2} \cdots s_{\beta_l}\ \}.$$

(2) Φ_{vw} is the disjoint union of Φ_w and $\Phi_v w$ whenever $l(vw) = l(v) + l(w)$;

(3) $\Phi_{w^{-1}} = -\Phi_w w^{-1}$;

(4) Φ^+ is the disjoint union of Φ_w and $\Phi_{w_0 w}$, where w_0 is the longest element of W.

Proof. Parts (1) and (2) are proved in Bourbaki (1981). Part (3) follows from the definition as follows:

$$-\Phi_w w^{-1} = -(\Phi^- w \cap \Phi^+) w^{-1} = \Phi^+ \cap \Phi^- w^{-1} = \Phi_{w^{-1}}.$$

Finally, Φ is the disjoint union of $\Phi^- w$ and $\Phi^+ w$, so Φ^+ is the disjoint union of $\Phi^+ \cap \Phi^- w = \Phi_w$ and $\Phi^+ \cap \Phi^+ w = \Phi^+ \cap \Phi^- w_0 w = \Phi_{w_0 w}$, and so we have (4). □

2.3. **Computation.** We now give a construction of root data suitable for computation. Take X and Y to be $\mathbb{Z}^d$ with $e_1, \ldots, e_d$ and $f_1, \ldots, f_d$ both the standard basis. A root datum is determined by a pair of $n \times d$ integer matrices A and B such that the rows of A are the simple roots and the rows of B are the corresponding coroots; ie. $A_{ij} = \langle \alpha_i, f_j \rangle$ and $B_{ij} = \langle e_j, \alpha_i^\star \rangle$. The Cartan matrix is $C = AB^t$.

Given a root β and a simple root α_i, knowledge of the Cartan matrix is sufficient to determine whether or not $\beta + \alpha_i$ is a root. Thus Φ^+ can be constructed recursively from the simple roots. The positive roots are constructed and stored in an order *compatible with height*, ie. $h(\alpha) < h(\beta)$ implies that $\alpha < \beta$. We extend this to a linear ordering on $\Phi^- \cup \{0\} \cup \Phi^+$ in the obvious way. The coroots are computed similarly.

3. Structure constants

We now briefly discuss the constants used to define semisimple Lie algebras and groups of Lie type, and efficient methods for computing them. See Carter (1965) or Riebeek (1998) for more details.

Fix a root datum $\mathcal{R} = (X, \Phi, Y, \Phi^\star)$. For every pair of linearly independent roots α and β, we define $p_{\alpha\beta}$ to be the largest integer such that $-p_{\alpha\beta}\alpha + \beta$ is a root and $q_{\alpha\beta}$ to be the largest integer such that $q_{\alpha\beta}\alpha + \beta$ is a root. The vectors

$$-p_{\alpha\beta}\alpha + \beta, \ldots, -\alpha + \beta,\ \beta,\ \alpha + \beta, \ldots, q_{\alpha\beta}\alpha + \beta$$

are all roots; they form the *chain* through β in the direction of α. Furthermore $\langle \beta, \alpha^\star \rangle = p_{\alpha\beta} - q_{\alpha\beta}$.

The semisimple Lie algebra with root system Φ has basis elements e_α, for α a root, and h_α, for α a simple root, satisfying the relations

$$[h_\alpha, h_\beta] = 0, \qquad [e_\alpha, h_\beta] = \langle \alpha, \beta^\star \rangle\, e_\alpha,$$

$$[e_{-\alpha}, e_\alpha] = \sum_{i=1}^{n} \langle \omega_i, \alpha^\star \rangle h_{\alpha_i}, \qquad [e_\alpha, e_\beta] = \begin{cases} N_{\alpha\beta}\, e_{\alpha+\beta} & \text{if } \alpha + \beta \in \Phi, \\ 0 & \text{otherwise.} \end{cases}$$

Chevalley (1955) showed that the constants $N_{\alpha\beta}$ may be chosen to have the form $\varepsilon_{\alpha\beta}(p_{\alpha\beta} + 1)$ where $\varepsilon_{\alpha\beta} = \pm 1$. This ensures that all the constants that appear in these relations are integral, so we can define a Lie algebra over an arbitrary field.

In order to multiply elements in the Steinberg presentation efficiently, we must compute the root chains and constants $\varepsilon_{\alpha\beta}$, $p_{\alpha\beta}$, $q_{\alpha\beta}$ and $N_{\alpha\beta}$ very rapidly. It would be wasteful to store them for all pairs of roots, so we precompute them only for pairs (α, β) with $0 < \alpha < \beta$ and $\alpha + \beta \in \Phi$. There are $O(n^3)$ such pairs, as can be shown by a case-by-case analysis for each of the classical types A_n, B_n, C_n, D_n. The $\varepsilon_{\alpha\beta}$ for such pairs can be computed by the cochain formula, which was developed for simply laced types by Frenkel and Kac (1980/81) and has been extended to all types by Rylands (2001); or alternatively by the method of extraspecial pairs (Carter, 1965, Section 4.2). A modification of the cochain formula has been implemented in both GAP (1999) and Magma (Bosma and Cannon, 1997) by Willem de Graaf. The root chains and other constants can be computed directly from the root system. Having stored the root chains and constants for all such pairs, we can rapidly compute them for an arbitrary pair of roots. For example, we

can compute $N_{\alpha\beta}$ using the algorithm of Gilkey and Seitz (1988), which we have implemented in Magma (Bosma and Cannon, 1997). We make the algorithm more efficient by avoiding the use of recursion.

The following constants are now easily computed: whenever $(i-1)\alpha+\beta$ is a root define

$$M_{\alpha\beta i} = \binom{p_{\alpha\beta}+i}{i}\varepsilon_{\beta\alpha}\cdots\varepsilon_{(i-1)\alpha+\beta,\alpha};$$

whenever $i\alpha+j\beta$ is a root define

$$C_{ij\alpha\beta} = \begin{cases} -M_{\alpha\beta i} & \text{for } j=1,\\ M_{\beta\alpha j} & \text{for } i=1,\\ -\frac{2}{3}M_{\alpha+\beta,\alpha,2} & \text{for } i=3, j=2,\\ -\frac{1}{3}M_{\alpha+\beta,\beta,2} & \text{for } i=2, j=3; \end{cases}$$

and for $p=p_{\alpha\beta}$ and $q=q_{\alpha\beta}$ define

$$\eta_{\alpha\beta} = (-1)^p \frac{\varepsilon_{\beta-p\alpha,\alpha}\cdots\varepsilon_{\beta-\alpha,\alpha}}{\varepsilon_{\beta-p\alpha,\alpha}\cdots\varepsilon_{\beta+(q-p-1)\alpha,\alpha}}.$$

Note that these constants are also integral, thus allowing the presentations given in the next section to be defined for an arbitrary field.

4. The Steinberg presentation

In this section we describe a presentation for the group of Lie type with root datum $\mathcal{R}=(X,\Phi,Y,\Phi^\star)$ over an arbitrary field $\mathbb{F}$. More details on this presentation can be found in, for example, Springer (1998).

4.1. **The presentation.** The Steinberg presentation for the group of Lie type $G_{\mathcal{R}}(\mathbb{F})$ has generators $x_\alpha(a)$, for α a root and $a\in\mathbb{F}$, and $y\otimes t$, for $y\in Y$ and $t\in\mathbb{F}^\times$. We also define auxiliary generators $n_\alpha = x_\alpha(1)x_{-\alpha}(-1)x_\alpha(1)$. The relations are

$$(y\otimes t)(y\otimes u) = y\otimes(tu), \tag{1}$$

$$(y\otimes t)(z\otimes t) = (y+z)\otimes t, \tag{2}$$

$$\alpha^\star\otimes t = x_\alpha(-1)x_{-\alpha}(1)x_\alpha(-1)\cdot x_\alpha(t)x_{-\alpha}(-t^{-1})x_\alpha(t) \tag{3}$$

$$(y\otimes t)^{n_\beta} = ys_\beta^\star\otimes t \tag{4}$$

$$x_\alpha(a)x_\alpha(b) = x_\alpha(a+b), \tag{5}$$

$$x_\alpha(a)^{x_\beta(b)} = x_\alpha(a)\prod_{i,j>0} x_{i\alpha+j\beta}(C_{ij\alpha\beta}a^ib^j), \tag{6}$$

$$x_\alpha(a)^{x_{-\alpha}(t)} = x_{-\alpha}(-t^2a)^{x_\alpha(t^{-1})} \tag{7}$$

where α and β are linearly independent roots, $y,z\in Y$, $a,b\in\mathbb{F}$ and $t,u\in\mathbb{F}^\times$. The product on the right hand side of (6) runs over roots of the form $i\alpha+j\beta$ (for i and j positive integers) in the order described in Subsection 2.3. Relation (7) is redundant except when the rank is one.

4.2. **Structure.** The group $G_{\mathcal{R}}(\mathbb{F})$ is linear algebraic, even though we have described it abstractly. We have not used the presentations described in Carter (1993) or Steinberg (1968) because they define groups which are not necessarily algebraic when $\mathbb{F}$ is not algebraically closed.

The following result follows from Section 11.1 of Carter (1965).

Theorem 4.1. *If $\mathcal{R}$ is irreducible, then $G_{\mathcal{R}}(\mathbb{F})'/Z(G_{\mathcal{R}}(\mathbb{F}))$ is a simple group, unless $\mathcal{R}$ has type A_1 with $|\mathbb{F}| \leq 3$, or type B_2 or G_2 with $|\mathbb{F}| = 2$.*

More generally, suppose $G = G_{\mathcal{R}}(\mathbb{F})$ where $\mathcal{R}$ does not contain a summand of type A_1 with $|\mathbb{F}| \leq 3$, or of type B_2 or G_2 with $|\mathbb{F}| = 2$. Then $G'/Z(G)$ is a direct sum of simple groups. At the top we have $G/G' = (Y \otimes \mathbb{F}^\star)/(\mathbb{Z}\Phi^\star \otimes \mathbb{F}^\star)$, which is isomorphic to a subgroup of $Y/\mathbb{Z}\Phi^\star$. At the bottom we have

$$Z(G) = \left\{ \prod_{i=1}^{n} f_i \otimes h_i \;\middle|\; \prod_{i=1}^{n} h_i^{\langle x, f_i \rangle} = 1 \text{ for all } x \text{ in } \mathbb{Z}\Phi \right\},$$

which is isomorphic to a subgroup of $X/\mathbb{Z}\Phi$ (Carter, 1972, page 198). In particular, if $\mathbb{F}$ is algebraically closed, then $G/G' \cong (\mathbb{F}^\times)^{d-n}$ and $Z(G) \cong X/\mathbb{Z}\Phi$.

We define the following important subgroups of $G_{\mathcal{R}}(\mathbb{F})$.

- The maximal torus H is the subgroup generated by the elements $y \otimes t$. It is isomorphic to the tensor product of the abelian groups Y and $\mathbb{F}^\times$.
- N is the subgroup generated by the maximal torus and the Weyl terms n_α; it is the normaliser of H, except in certain cases where the field is small. For w in the Weyl group W, take a reduced expression $w = s_{\beta_1} \cdots s_{\beta_l}$ and set $\dot{w} = n_{\beta_1} \cdots n_{\beta_l}$; this is well defined by Proposition 9.3.2 of Springer (1998). There is an isomorphism between N/H and W given by $H\dot{w} \leftrightarrow w$.
- The unipotent subgroup U is generated by the elements $x_\alpha(a)$ for α a positive root. Note that U is a nilpotent group and if $\mathbb{F}$ has characteristic p it is a p-group. For w in W, define U_w to be the subgroup of U generated by the elements $x_\alpha(a)$ for α in Φ_w.
- The Borel subgroup $B = HU$. Its double cosets correspond to the elements of the Weyl group by $B\dot{w}B \leftrightarrow w$.

4.3. **Example: Type A_n.** Suppose $\mathcal{R}$ is a semisimple root datum of type A_n. Then the quotient $X/\mathbb{Z}\Phi$ is a subgroup of the cyclic group of order $n+1$; suppose it is the subgroup of size d for some d dividing $n+1$. The group of Lie type $G_{\mathcal{R}}(\mathbb{F}) \cong \mathrm{SL}_{n+1,d}(\mathbb{F})/Z_d$, where

$$\mathrm{SL}_{n+1,d}(\mathbb{F}) = \{X \in \mathrm{GL}_{n+1}(\mathbb{F}) \mid \det(X)^d = 1\} \quad \text{and}$$
$$Z_d = \{tI_{n+1} \mid t^d = 1\}.$$

In particular, for $d = 1$ we get $\mathrm{SL}_{n+1}(\mathbb{F})$ and for $d = n+1$ we get $\mathrm{PGL}_{n+1}(\mathbb{F})$. Note that some properties of these groups are dependent on the field. For example, if $\mathbb{F}$ is algebraically closed, then $\mathrm{PGL}_{n+1}(\mathbb{F}) = \mathrm{PSL}_{n+1}(\mathbb{F})$ which is simple as an abstract group. In general $\mathrm{PGL}_{n+1}(\mathbb{F})/\mathrm{PSL}_{n+1}(\mathbb{F}) \cong \mathbb{F}^\times/(\mathbb{F}^\times)^{n+1}$, and $\mathrm{PSL}_{n+1}(\mathbb{F})$ may not be algebraic.

5. Arithmetic and data structures

We now describe algorithms for element arithmetic in the group of Lie type $G = G_{\mathcal{R}}(\mathbb{F})$. A normal form for elements is a very useful tool for doing computations in a group. In the case of permutation groups, for example, we get a normal form

from a base and strong generating set. In our case, we use the Bruhat decomposition (Carter, 1965, Corollary 8.4.4): each $g \in G$, can be expressed uniquely in the form

$$g = uh\dot{w}u',$$

where $u \in U$, $h \in H$, $w \in W$, and $u' \in U_w$.

5.1. **Actions.** The generators of G are of three types: unipotent terms $x_\alpha(t)$, Weyl terms n_α, and torus terms $y \otimes t$. The following equations describe how these generators act on each other, thus allowing us to exchange unipotent, Weyl and torus terms in a word. We also give equations for unipotent terms with negative roots. It is convenient to introduce the notation $h_\alpha(t) = \alpha^\star \otimes t$.

- The action of the Weyl term on the maximal torus:

$$(y \otimes t)^{n_\alpha} = ys_\alpha^\star \otimes t \tag{8}$$

$$^{n_\alpha}(y \otimes t) = ys_\alpha^\star \otimes t \tag{9}$$

- The action of the Weyl term on the unipotent subgroup:

$$x_\beta(a)^{n_\alpha} = x_{\beta s_\alpha}(\eta_{\alpha,\beta} a) \tag{10}$$

$$^{n_\alpha}x_\beta(a) = x_{\beta s_\alpha}(\eta_{\alpha,\beta}(-1)^{\langle \alpha, \beta^\star \rangle} a) \tag{11}$$

- The action of the maximal torus on the unipotent subgroup:

$$x_\beta(a)^{y\otimes t} = x_\beta(t^{\langle \beta, y \rangle} a) \tag{12}$$

$$^{y\otimes t}x_\beta(a) = x_\beta(t^{-\langle \beta, y \rangle} a) \tag{13}$$

- Negative roots:

$$x_{-\alpha}(t) = x_\alpha(t^{-1}) n_\alpha h_\alpha(-t^{-1}) x_\alpha(t^{-1}) \tag{14}$$

$$x_\alpha(t) n_\alpha = x_{-\alpha}(t^{-1}) h_\alpha(-t^{-1}) x_\alpha(-t^{-1}) \tag{15}$$

We now discuss computation with torus elements, Weyl group representatives and unipotent elements in turn; before putting it all together in Subsection 5.5.

5.2. **Torus elements.** The torus is $H = Y \otimes \mathbb{F}^\times$. Recall that Y has a basis $f_1, \ldots, f_d$, so every element of $Y \otimes \mathbb{F}^\times$ can be written uniquely in the form $\prod_{i=1}^d f_i \otimes h_i$ where $h_i \in \mathbb{F}^\times$. This element can be represented on the computer by the vector $(h_1, \ldots, h_d)$. Multiplication and inversion are just coordinatewise operations.

Given an automorphism of Y whose matrix with respect to $f_1, \ldots, f_d$ is M, we get an induced automorphism of $Y \otimes \mathbb{F}^\times$

$$\prod_{i=1}^d f_i \otimes h_i \mapsto \prod_{i=1}^d \left(\sum_{j=1}^d M_{ij} f_j \right) \otimes h_i \;=\; \prod_{j=1}^d f_j \otimes \left(\prod_{i=1}^d h_i^{M_{ij}} \right).$$

Using equation (8), we can compute the action of n_α on a torus element by taking M to be the matrix of $s_\alpha^\star$.

A torus element is in the centre of G if it is of the form $\prod_{i=1}^n f_i \otimes h_i$ with $\prod_{i=1}^n h_i^{\langle x, f_i \rangle} = 1$ for all $x \in \mathbb{Z}\Phi$. This is equivalent to

$$1 = \prod_{i=1}^d h_i^{\langle \alpha_j, f_i \rangle} = \prod_{i=1}^d h_i^{A_{ji}}$$

for $j = 1, \ldots, n$.

5.3. **Weyl group representatives.** Recall that the Weyl group W is a Coxeter group and $\dot{w} = n_{\beta_1} \cdots n_{\beta_l}$ where w has reduced expression $s_{\beta_1} \cdots s_{\beta_l}$. The following equations are useful for computation:

$$n_\beta{}^{n_\alpha} = n_{\beta s_\alpha} h_{\beta s_\alpha}(\eta_{\alpha,\beta}), \tag{16}$$

$$n_\alpha{}^{-1} = n_{-\alpha} = h_\alpha(-1) n_\alpha = n_\alpha h_\alpha(-1), \tag{17}$$

$$n_\alpha{}^2 = h_\alpha(-1). \tag{18}$$

We compute products by repeated use of the formula

$$\dot{w} n_\alpha = \begin{cases} (ws_\alpha)^\cdot & \text{if } \alpha \notin \Phi_w,\ \alpha \text{ simple}, \\ h_{\alpha w^{-1}}(-1)\,(ws_\alpha)^\cdot & \text{if } \alpha \in \Phi_w,\ \alpha \text{ simple}. \end{cases}$$

The inverse of $\dot{w}$ is computed as follows, using Lemma 2.1(1):

$$\begin{aligned} \dot{w}^{-1} &= \dot{s}_{\beta_l}^{-1} \cdots \dot{s}_{\beta_2}^{-1} \dot{s}_{\beta_1}^{-1} \\ &= n_{\beta_l} h_{\beta_l}(-1) \cdots n_{\beta_2} h_{\beta_2}(-1) \cdot n_{\beta_1} h_{\beta_1}(-1) \\ &= (n_{\beta_l} \cdots n_{\beta_1}) h_{\beta_l s_{\beta_{l-1}} \cdots s_{\beta_1}}(-1) \cdots h_{\beta_2 s_{\beta_1}}(-1) h_{\beta_1}(-1) \\ &= (w^{-1})^\cdot \prod_{\alpha \in \Phi_{w^{-1}}} h_\alpha(-1). \end{aligned}$$

Note that when α is a nonsimple root, $n_\alpha = h\dot{s}_\alpha$ where the torus element h can be computed using (16) and the fact that s_α is conjugate to a simple reflection of W.

5.4. **The unipotent subgroup.** The unipotent subgroup U is generated by the elements $x_\alpha(a)$ for α a positive root and $a \in \mathbb{F}$. It is shown in Carter (1972) that every element of U can be written uniquely in the form $\prod_{\alpha \in \Phi^+} x_\alpha(a_\alpha)$ where the product is in the order described in Subsection 2.3. This proof is constructive and proceeds by collection using the commutator formulas (5). Computationally we use collection from the left to put a unipotent element into this normal form. This is generally the most efficient known collection algorithm (Leedham-Green and Soicher, 1990) and our testing indicates that it is most efficient in this case as well. We invert a unipotent element by reversing the order of the terms and replacing the field elements by their negatives, then doing collection to put the terms back in the right order.

Given an Weyl group element w, Carter (1972) also proves that we can collect any unipotent element into the form

$$\prod_{\alpha \in \Phi^+ \setminus \Phi_w} x_\alpha(a_\alpha) \cdot \prod_{\alpha \in \Phi_w} x_\alpha(a_\alpha). \tag{19}$$

Note that $\prod_{\alpha \in \Phi_w} x_\alpha(a_\alpha)$ is a general element of the subgroup U_w. This is used to normalise an element of the form $uh\dot{w}u'$ with u' in U but not necessarily in U_w: put u' in the form (19), then exchange the first part with $h\dot{w}$ using (10) and (12) and multiply u by the result.

Finally note that we can easily collect all terms corresponding to a fixed root α, so every unipotent element can be put in the form

$$x_\alpha(a) \cdot u' \text{ where no term of the form } x_\alpha(a') \text{ appears in } u'. \tag{20}$$

write u' in the form $x_\alpha(t) \cdot u'$ of (20)	$uh\dot{w}x_\alpha(t)u'n_\alpha$
exchange n_α and u' by (10)	$uh\dot{w}x_\alpha(t)n_\alpha u'$
if $t \neq 0$ and $\alpha \in \Phi_w$ (ie. $gn_\alpha \in B\dot{w}B$) then	
apply (15) to $x_\alpha(t)n_\alpha$	$uh\dot{w}x_{-\alpha}(t^{-1})h_\alpha(-t^{-1})x_\alpha(-t^{-1})u'$
replace u' by $x_\alpha(-t^{-1})u'$	$uh\dot{w}x_{-\alpha}(t^{-1})h_\alpha(-t^{-1})u'$
exchange $x_{-\alpha}(-t^{-1})$ with $h\dot{w}$ by (11) and (13)	
and multiply u by the result	$uh\dot{w}h_\alpha(-t^{-1})u'$
exchange $h_\alpha(-t^{-1})$ with $\dot{w}$ by (9)	
and multiply h by the result	$uh\dot{w}u'$
else ($gn_\alpha \in B\dot{w}n_\alpha B$)	
if $t \neq 0$ then	
exchange $x_\alpha(t)$ with $h\dot{w}$ by (11) and (13)	
and multiply u by the result	$uh\dot{w}n_\alpha u'$
end if	
multiply w by s_α	
if $\alpha \in \Phi_w$ then	
multiply h by $h_{\alpha w^{-1}}(-1)$	
end if	$uh\dot{w}u'$
end if	

Algorithm 1. Multiplication by a Weyl term

5.5. **Normalisation and arithmetic.** Recall that we can exchange unipotent, Weyl and torus terms in a word using the equations in Subsection 5.1. Our goal is to efficiently use these equations to put a word into Bruhat normal form. It suffices to give an algorithm for normalising the product of a normalised element by a unipotent element, torus element or Weyl term. Note that we assume that all the terms $x_\alpha(t)$ which appear in our word have α positive—any negative roots that appear in the original word can be eliminated with equation (14). Much of the difficulty in this algorithm is ensuring that the roots remain positive when we exchange unipotent terms with Weyl terms.

Suppose $g = uh\dot{w}u'$ is in normal form. To multiply g by a torus element, exchange the torus element with $\dot{w}u'$ using (12) and (9), then multiply h by the result. To multiply g by a unipotent element, multiply u' by the unipotent element and normalise $uh\dot{w}u'$ as in the previous subsection.

Finally consider the more complicated case of multiplication by a Weyl term n_α. Recall that

$$gn_\alpha = uh\dot{w}u'n_\alpha \in B\dot{w}Bn_\alpha \subseteq B\dot{w}n_\alpha B \cup B\dot{w}B,$$

where the union is disjoint. This gives us the two main cases of Algorithm 1. In order to clarify the pseudocode we write the word we are working on to the right, but note that the values of the symbols u, h, w and u' change from line to line.

It is now easy to normalise the product of two normalised words. We invert a normalised word by inverting each term and reversing their order, then swapping h and $\dot{w}$ and normalising the unipotent parts as in Subsection 5.4. We can do conjugation more efficiently using the formulas of Subsection 5.1 directly rather than computing the product $y^{-1}xy$, but the result still needs to be normalised. Commutators are computed with one inversion, one conjugation and one multiplication: $[x, y] = x^{-1} \cdot x^y$.

Note that the most time consuming part of these arithmetic operations is collection of unipotent elements. Collection is also the only part of this algorithm which is not known to run in polynomial time. In Subsection 7.7, we show that there is a provably polynomial time alternative to collection, which nevertheless seems to be slower in practice.

6. Computing linear representations

Computing an irreducible highest weight representation for a group of Lie type is fairly straightforward since the representations for semisimple Lie algebras can already be computed by the algorithm of de Graaf (2001). Let G be a group of Lie type, so the derived group G' is a semisimple group. Let L be the integral Lie algebra corresponding to G' and let $\rho : L \to \mathfrak{gl}(V)$ be the integral irreducible highest weight representation with highest weight λ in X. We define $x_\alpha(T)$ as $\exp(T\rho(e_\alpha)) \in \mathrm{End}_{\mathbb{Z}[T]}(V)$, where T is an indeterminate. The matrix for $x_\alpha(t)$ for $t \in \mathbb{F}$ is simply gotten by substitution; this acts on the $\mathbb{F}$-vector space $V \otimes_{\mathbb{Z}} \mathbb{F}$. We also need matrices for the elements of the torus $Y \otimes \mathbb{F}^\times$ which are not contained in $\mathbb{Z}\Phi^\star \otimes \mathbb{F}^\times$—this requires a linear algebra calculation the details of which we omit.

This algorithm runs in polynomial time in the coefficients of λ for a fixed group G. It is not possible for the algorithm to be polynomial in the rank of G and the coefficients of λ, since the dimension of the irreducible highest weight module is not polynomial in these parameters.

The practical efficiency of this computation can be improved by noticing that the matrices corresponding to terms in the Steinberg presentation are sparse. So rather than multiplying on the left or right by the matrices, we use the corresponding row or column operations. This observation motivates the algorithm of the next section.

7. Generalised row and column reduction

We now consider the converse problem of converting from a representation to the Steinberg presentation. Together with the previous section, this allows us to transfer between representations and the Steinberg presentation, so we can use whichever is most appropriate for the computation at hand.

Our algorithm is a generalisation of row and column reduction or, more specifically, the LUP algorithm. It works for any nontrivial quotient of a highest weight representation. In many cases it can be extended to an arbitrary representation. First note that we can reduce to the case of an irreducible representation by working our way down a composition series of the module. If $\mathbb{F}$ is algebraically closed or finite, then every irreducible module is a quotient of a highest weight module.

We assume that G is a group of Lie type $G_{\mathcal{R}}(\mathbb{F})$ and that $\rho : G \to \mathrm{GL}(V)$ is a nontrivial quotient of a highest weight representation. We consider V to be a G-module by $vg := v\rho(g)$. Let A be a matrix in $\mathrm{GL}(V)$; to simplify our exposition we assume that $A \in \rho(G)$ (in fact, our algorithm is easily converted into a membership test for $\rho(G)$). Our aim is to find $g \in G$ so that $A = \rho(g)$, where g is given as a word in Bruhat normal form.

7.1. **The case of $\mathrm{SL}_2(\alpha)$.** Given a root α, define $\mathrm{SL}_2(\alpha)$ to be the subgroup of G generated by the elements $x_\alpha(a)$ and $x_{-\alpha}(a)$ for all $a \in \mathbb{F}$. This subgroup is isomorphic to a quotient of $\mathrm{SL}_2(\mathbb{F})$. We define an action of $\mathrm{SL}_2(\alpha)$ on the polynomial

if $vA \cdot vn_\alpha \neq 0$ then
  let $w = s_\alpha$
  let $c = (-1)^m (vA \cdot v)(vA \cdot vn_\alpha)^{-1}$
  take b to be the mth root of c for which $vn_\alpha A x_\alpha(b)^{-1} \cdot vn_\alpha = 0$
  let $A = A\rho(n_\alpha x_\alpha(b))^{-1}$
else
  set $w = 1$, $b = 0$
end if
let $c = (-1)^m (vn_\alpha A \cdot v)(vA \cdot v)^{-1}$
take a to be the mth root of c for which $vn_\alpha x_\alpha(a)^{-1} A \cdot v = 0$
take t to be the mth root of $vg \cdot v$ for which $A = \rho(h_\alpha(t))$
return $g = x_\alpha(a) h_\alpha(t) \dot{w} x_\alpha(b)$

ALGORITHM 2. The $\mathrm{SL}_2(\alpha)$ algorithm

algebra $\mathbb{F}[x, y]$ by

$$\begin{aligned} x_\alpha(a): &\quad x \mapsto x, && y \mapsto ax + y; \\ x_{-\alpha}(a): &\quad x \mapsto x + ay, && y \mapsto y; \\ n_\alpha: &\quad x \mapsto -y, && y \mapsto x; \\ h_\alpha(t): &\quad x \mapsto tx, && y \mapsto t^{-1} y. \end{aligned}$$

The highest weight module for $\mathrm{SL}_2(\alpha)$ of weight m is just the submodule of $\mathbb{F}[x, y]$ consisting of homogeneous polynomials of degree m; we denote this module by $V(m)$. Note that, since $\mathrm{SL}_2(\alpha)$ is a quotient of $\mathrm{SL}_2(\mathbb{F})$, this may be a projective representation—for our purposes however this is not a problem. If $\mathbb{F}$ has characteristic 0, $V(m)$ is irreducible. If $\mathbb{F}$ has characteristic p, then we get an irreducible module by quotienting out the submodule spanned by the terms $x^i y^{m-i}$ for which p divides $\binom{m}{i}$.

Every irreducible $\mathrm{SL}_2(\alpha)$-module M is isomorphic to a nontrivial quotient of $V(m)$ for some m. First we need to compute the value of m. Let v be a nonzero fixed vector of $x_\alpha(1)$. Then v is the image of x^m and we can compute m with the formula $vh_\alpha(t) = t^m v$. If $\mathbb{F}$ is finite of size q, then we can assume that $m < q$ by the Steinberg tensor product theorem (Steinberg, 1963) and so m is the smallest positive integer satisfying this formula for t a primitive element of $\mathbb{F}$. We also know that $(-1)^m vn_\alpha$ is the image of y^m and the images of the subspaces $\mathbb{F}x^i y^{m-i}$ are found by taking eigenspaces of $h_\alpha(t)$. Hence we can compute an inner product $\cdot$ on V so that these one-dimensional subspaces are orthogonal, and x^m and y^m have norm one.

Given a matrix $A \in \mathrm{GL}(V)$, we want to find $a, b \in \mathbb{F}$, $t \in \mathbb{F}^\times$ and $w \in W = \{1, s_\alpha\}$ such that $A = \rho(x_\alpha(a) h_\alpha(t) \dot{w} x_\alpha(b))$. By an easy computation, we get

$$\begin{aligned} &x^m g \cdot y^m = (-t)^m, \quad x^m g \cdot x^m = (-bt)^m; \\ &x^m h \cdot y^m = 0, \qquad x^m h \cdot x^m = t^m, \qquad y^m h \cdot x^m = (at)^m; \end{aligned}$$

where $g = x_\alpha(a) h_\alpha(t) n_\alpha x_\alpha(b)$ and $h = x_\alpha(a) h_\alpha(t)$. Hence we have Algorithm 2, using the fact that v is the image of x^m and $(-1)^m vn_\alpha$ is the image of y^m.

7.2. **Computing weights and weight spaces.** Given $\mu \in X$, we define

$$V_\mu = \{v \in V \mid v(y \otimes t) = t^{\langle \mu, y \rangle} v \text{ for all } y \in Y,\ t \in \mathbb{F}^\times\}.$$

Then $V = \bigoplus_\mu V_\mu$ and we call μ a *weight* of V if V_μ is nontrivial. The *dominance order* for weights is defined by $\mu \preceq \nu$ if $\mu - \nu = \sum_{i=1}^n m_i\alpha_i$ where every $m_i \geq 0$; we write $\mu \prec \nu$ when $\mu \preceq \nu$ and $\mu \neq \nu$. Since V is a nontrivial quotient of a highest weight representation, it has a dominance-highest weight λ and V_λ is one dimensional. In many applications, λ and V_λ are assumed to be known. However they can be computed if necessary: V_λ is the intersection of the fixed spaces of the matrices $\rho(x_\alpha(1))$ for α a simple root. The highest weight can be computed using the formula $\lambda = \sum_{i=1}^n \langle \lambda, \alpha_i^\star\rangle \omega_i$, and the fact that the $\mathrm{SL}_2(\alpha_i)$-submodule generated by V_λ is a nontrivial quotient of $V(m)$ with $m = \langle \lambda, \alpha_i^\star\rangle$.

Fix a nonzero vector v_λ in V_λ. Write Ω for the orbit of λ under the action of W; for each $\mu \in \Omega$, choose some $w \in W$ so that $\mu = \lambda w$ and fix $v_\mu = v_\lambda \dot{w}$. Note that $V_\mu = \mathbb{F}v_\mu$ and v_μ is well defined up to a sign. We can compute V_μ for a weight $\mu \notin \Omega$ by taking eigenspaces of torus elements. This allows us to compute an inner product $\centerdot$ on V under which the decomposition $V = \bigoplus_\mu V_\mu$ is orthogonal and each v_μ for $\mu \in \Omega$ has norm one.

We use the following basic facts about the action of G:

$$V_\mu x_\alpha(t) \subseteq \bigoplus_{\nu \succeq \mu} V_\nu \quad \text{for } \alpha \in \Phi^+ \tag{21}$$

$$V_\mu \dot{w} = V_{\mu w} \tag{22}$$

$$V_\mu H = V_\mu \tag{23}$$

7.3. **The inductive step.** In this section we show how to reduce from G to a reductive subgroup of smaller semisimple rank.

First we define a dual action of G on V. By Theorem 29 of Steinberg (1968), there is an automorphism of the semisimple group G' given by $x_\alpha(t) \mapsto x_{-\alpha}(\varepsilon_\alpha t)$ for some constants ε_α. We define an antiautomorphism δ on G by

$$x_\alpha(t)^\delta = x_{-\alpha}(\varepsilon_\alpha t)^{-1} = x_{-\alpha}(-\varepsilon_\alpha t), \quad \text{and}$$
$$(y \otimes t)^\delta = y \otimes t.$$

It is easily shown that $H\dot{w}^\delta = H\dot{w}^{-1}$. We get a left action of G on V by $gv = vg^\delta$. We also choose our constants so that $\delta^2 = 1$. Note that if the longest word w_0 acts as -1 on X, then V is selfdual and δ can be taken to be $g^\delta = (g^{\dot{w}_0})^{-1}$.

Since λ is a dominant weight, its stabiliser W_λ is actually the standard parabolic subgroup W_J with $J = \{s \mid \langle \lambda, \alpha_s^\star\rangle = 0\}$. Within G we have opposite parabolic subgroups $P_J = B\dot{W}_J B$ and $P_J^\delta = B^\delta \dot{W}_J B^\delta$. The Levi complement $L_J = P_J \cap P_J^\delta$ is reductive with root datum $(X, \Phi_\lambda, Y, \Phi_\lambda^\star)$, where

$$\Phi_\lambda = \{\alpha \in \Phi \mid \langle \lambda, \alpha^\star\rangle = 0\} = \Phi_J$$

and $\Phi_\lambda^\star = \{\alpha^\star \mid \alpha \in \Phi_\lambda\}$. In particular, the semisimple rank of L_J is strictly smaller than the semisimple rank of G and the Weyl group of L_J is $W_J = W_\lambda$.

Proposition 7.1. *Let g be an element of G. Then $g \in L_J$ if, and only if, $V_\lambda g = V_\lambda$ and $gV_\lambda = V_\lambda$.*

Proof. First we show that $g \in P_J$ if, and only if, $V_\lambda g = V_\lambda$. Since λ is the highest weight, $V_\lambda B = V_\lambda$ and so $V_\lambda B\dot{w}B = V_{\lambda w}B = \sum_{\mu \succeq \lambda w} V_\mu$ by (21) and (23). Clearly this is equal to V_λ exactly when $\lambda w = \lambda$, i.e. $w \in W_J$.

On the other hand, $gV_\lambda = V_\lambda$ means $V_\lambda = V_\lambda g^\delta$ which is equivalent to $g^\delta \in P_J$, i.e. $g \in P_J^\delta$. Hence we are done. □

Any element w of W can be decomposed in the form $w_J w^J$ where $w_J \in W_J$ and w^J is the unique W_J-reduced element of W such that $\lambda w^J = \lambda w$. We write W^J for the set of all W_J-reduced elements of W. By Lemma 2.1(2), Φ_w is a disjoint union of $\Phi_{w_J w^J}$ and Φ_{w^J}, and so we get a unique decomposition $U_w = (U_{w_J})^{\dot{w}^J} U_{w^J}$. We are particularly interested in the factorisation $w_0 = w_{0J} w_0^J$ of the longest word in W. It is easily seen that w_{0J} is the longest word of W_J. Since $w_{0J} = w_0 (w_0^J)^{-1}$, Lemma 2.1(4) implies that Φ^+ is the disjoint union of $\Phi_{(w_0^J)^{-1}}$ and $\Phi_{w_{0J}}$ so we get a unique decomposition $U_w = U_{(w_0^J)^{-1}} U_{w_{0J}}$. We can now modify the Bruhat decomposition as follows:

$$\begin{aligned} G &= \bigcup_{w \in W} U H \dot{w} U_w \\ &= \bigcup_{w_J \in W_J, w^J \in W^J} U_{(w_0^J)^{-1}} U_{w_{0J}} H \dot{w}_J \dot{w}^J U_{w_J}^{\dot{w}^J} U_{w^J} \\ &= \bigcup_{w^J \in W^J} U_{(w_0^J)^{-1}} \left(\bigcup_{w_J \in W_J} U_{w_{0J}} H \dot{w}_J U_{w_J} \right) \dot{w}^J U_{w^J} \\ &= \bigcup_{w^J \in W^J} U_{(w_0^J)^{-1}} L_J \dot{w}^J U_{w^J}. \end{aligned}$$

Recall that we are given an element $A \in \rho(G)$ and we wish to find a preimage $g \in G$ so that $\rho(g) = A$. The first step is to find the value of w^J in this decomposition of g; then we can find $u \in U_{(w_0^J)^{-1}}$ and $u' \in U_{w^J}$ so that $A \in \rho(u L_J \dot{w}^J u')$. The problem is then reduced to finding a word for $\rho(u)^{-1} A \rho(\dot{w}^J u')^{-1}$ in the reductive subgroup L_J.

Since w^J is the unique W_J-reduced element of W with $\lambda w^J = \lambda w$, it can be found using the following lemma.

Lemma 7.2. *Let g be an element of the double coset $B\dot{w}B$ of G. If μ is the minimal element of Ω such that $v_\lambda g \cdot v_\mu \neq 0$, then $\lambda w = \mu$.*

Proof. This follows from the facts that $V_\lambda B \dot{w} B \subseteq \sum_{\mu \geq \lambda w} V_\mu$ and $v_\lambda g \cdot v_{\lambda w} \neq 0$. □

We now know that $A = \rho(g)$ with $g \in U_{(w_0^J)^{-1}} L_J \dot{w}^J U_{w^J}$ for fixed w^J. Write $g = uh\dot{w}^J u'$ where $u \in U_{(w_0^J)^{-1}}$, $h \in L_J$, and $u' = \prod_{\alpha \in \Phi_{w^J}} x_\alpha(a_\alpha) \in U_{w^J}$. Let γ be a dominance minimal root in Φ_{w^J} with $a_\gamma \neq 0$. Using (20) we can rewrite u' in the form $x_\gamma(a_\gamma) \prod_{\alpha \in \Phi_{w^J}, \alpha \not\trianglelefteq \gamma} x_\alpha(a_\alpha)$, with different values a_α but a_γ still nonzero. As in Subsection 7.1, the $\mathrm{SL}_2(\gamma)$-module generated by v_λ can be identified with a quotient of $V(m)$ with $m = \langle \mu, \gamma^\star \rangle$, the vector v_λ with the image of x^m, and $(-1)^m v_\lambda n_\mu$ with the image of y^m. Now

$$\begin{aligned} v_\lambda A = v_\lambda g &= v_\lambda u h \dot{w}^J u' = v_\lambda h \dot{w}^J u' \in \mathbb{F}^\times v_\mu u' \\ &= \mathbb{F}^\times v_\mu x_\gamma(a_\gamma) \prod_{\alpha \in \Phi_{w^J}, \alpha \not\trianglelefteq \gamma} x_\alpha(a_\alpha) \\ &= \mathbb{F}^\times (x + a_\gamma y)^m \prod_{\alpha \in \Phi_{w^J}, \alpha \not\trianglelefteq \gamma} x_\alpha(a_\alpha) \\ &\subseteq \mathbb{F}^\times (v_\mu + a_\gamma x^{m-1} y + \cdots + a_\gamma^m (-1)^m v_\mu n_\gamma) + \sum_{\nu \not\trianglelefteq \mu s_\gamma} V_\nu \end{aligned}$$

The final line follows from the fact that the only weights that can appear are of the form $\nu = \mu - i\gamma + \xi$ where $i = 0, \ldots, m$ and ξ is a nonzero linear combination of roots $\alpha \in \Phi_{w^J}$, $\alpha \not\perp \gamma$ with nonnegative coefficients. It is easily seen that such weights satisfy $\nu \not\perp \mu - m\gamma = \mu s_\gamma$. We can now compute a_γ as follows: take

$$c = (-1)^m (v_\lambda A \cdot v_\mu n_\gamma)(v_\lambda A \cdot v_\mu)^{-1},$$

then a_γ is the mth root of c with the property that $v_\lambda A x_\gamma(a_\gamma)^{-1} \cdot v_\mu n_\gamma = 0$. By repeating this process for all the roots in Φ_{w^J} in an order compatible with the dominance order, we can compute the value of u'. It is easily seen that the order $<$ of Subsection 2.3 has the required property.

We now have $g = uh\dot{w}^J$ and we wish to find $u \in U_{(w_0^J)^{-1}}$. This is similar to the preceding case: write $u = \left(\prod_{\beta \in \Phi_{(w_0^J)^{-1}}, \beta \not\perp \gamma} x_\beta(b_\beta)\right) x_\gamma(b_\gamma)$ and then

$$\begin{aligned} Av_\mu = gv_\mu = uh\dot{w}^J v_\mu &\in \mathbb{F}^\times u v_\lambda = \mathbb{F}^\times v_\lambda u^\delta \\ &= \mathbb{F}^\times v_\lambda x_\gamma(-\varepsilon_\gamma b_\gamma) \prod_{\beta \in \Phi_{w^J}, \alpha \not\perp \gamma} x_\beta(-\varepsilon_\beta b_\beta) \\ &\subseteq \mathbb{F}^\times (v_\lambda + \cdots + (\varepsilon_\gamma b_\gamma)^m v_\lambda n_\gamma) + \sum_{\nu \not\perp \mu s_\gamma} V_\nu \end{aligned}$$

where $m = \langle \mu, \gamma^\star \rangle$. Hence we compute b_γ by taking

$$c = (-\varepsilon_\gamma)^m (v_\lambda n_\gamma \cdot Av_\mu)(v_\lambda A \cdot v_\mu)^{-1},$$

and computing the m root b_γ of c such that $v_\lambda n_\gamma \cdot x_\gamma(b_\gamma)^{-1} Av_\mu = 0$.

We now have all the ingredients required for our algorithm, which we describe in detail in the next two subsections.

7.4. **A base for the Weyl group.** By iterating the procedure of the previous section, we get a sequence of reductive subgroups

$$G = L_{J_1} \geq L_{J_2} \geq \cdots \geq L_{J_{k+1}} = H$$

where

$$\{1, \ldots, n\} = J_1 \supseteq J_2 \supseteq \cdots \supseteq J_{k+1} = \emptyset.$$

Note that this chain of subgroups depends only on G, not on the particular matrix A. Hence the subsets J_i and the highest weight for each L_{J_i} can be computed beforehand. This is achieved using the following theorem.

Theorem 7.3. *Let G be a reductive group with highest weight representation V. Consider the Weyl group as a permutation group on the weights of V. Then W has a base $\lambda_1, \ldots, \lambda_k$ with the following properties:*

(1) $W_{\lambda_1, \ldots, \lambda_{i-1}} = W_{J_i}$ for some $J_i \subseteq \{1, \ldots, n\}$; and
(2) λ_i is a highest weight for $L_{J_i} = B\dot{W}_{J_i}B$ acting on V.

Proof. We proceed by induction on i. Suppose that $\lambda_1, \ldots, \lambda_{i-1}$ satisfy properties (1) and (2) with $J_i \neq \emptyset$. We prove the existence of λ_i and J_{i+1} strictly contained in J_i satisfying the same properties. Choose some $j < i$ and some $r \in J_i$ so that $\{s \in J_i \mid \langle \lambda_j s_{\alpha_r}, \alpha_s^\star \rangle = 0\}$ is properly contained in J_i. Let $\lambda_i = \lambda_j s_{\alpha_r}$. By adding sufficiently large scalar multiples of $\lambda_1, \ldots, \lambda_{i-1}$ to λ_i we get a dominant weight

$$\lambda_i' = \sum_{l<i} a_l \lambda_l + \lambda_i = \sum_{l<i} a_l \lambda_l + \lambda_j - \langle \lambda_j, \alpha_r^\star \rangle \alpha_r$$

let $\lambda_1 = \lambda$, $J_1 = \{1, \ldots, n\}$, $J_2 = \{s \in J_1 \mid \langle \lambda_1, \alpha_s^\star \rangle = 0\}$
let $i = 2$, $j = 1$.
repeat
  if $\exists r \in J_j - J_i$ such that $\{s \in J_i \mid \langle \lambda_j s_r, \alpha_s^\star \rangle = 0\} \subsetneq J_i$ then
    set $\lambda_i = \lambda_j s_r$, $J_{i+1} = \{s \in J_i \mid \langle \alpha_r, \alpha_s^\star \rangle = 0\}$
    increment i by 1; set j to $i - 1$
  else
    decrement j by 1.
  end if.
until J_i is empty.
return the λ_i and J_i.

ALGORITHM 3. Computing a base of weights

Note that for $w \in W_{J_i} = W_{\lambda_1,\ldots,\lambda_{i-1}}$, we have $\lambda_i' w = \lambda_i'$ iff $\lambda_i w = \lambda_i$ iff $\alpha_r w = \alpha_r$. Hence for $s \in J_i$, we have $\langle \lambda_i', \alpha_s^\star \rangle = 0$ iff $\langle \alpha_r, \alpha_s^\star \rangle = 0$. Since λ_i' is a dominant weight $W_{\lambda_i'} = W_{\{s \mid \langle \lambda_i', \alpha_s^\star \rangle = 0\}}$, so

$$\begin{aligned} W_{\lambda_1,\ldots,\lambda_i} &= W_{\lambda_1,\ldots,\lambda_{i-1}} \cap W_{\lambda_i'} \\ &= W_{J_i} \cap W_{\{s \mid \langle \lambda_i', \alpha_s^\star \rangle = 0\}} \\ &= W_{\{s \in J_i \mid \langle \alpha_r, \alpha_s^\star \rangle = 0\}}. \end{aligned}$$

We now take $J_{i+1} = \{s \in J_i \mid \langle \alpha_r, \alpha_s^\star \rangle = 0\}$ and we have property (1).

Recall that the highest weight of a highest weight module is the unique nonzero dominant weight for that module. Hence, if λ_i is a weight for the G-module V, then it is a highest weight for the L_{J_i}-module V iff $\langle \lambda_i, \alpha_s^\star \rangle \geq 0$ for all $s \in J_i$. Suppose $s \in J_i$. Then

$$\langle \lambda_i, \alpha_s^\star \rangle = \langle \lambda_j s_{\alpha_r}, \alpha_s^\star \rangle = \langle \lambda_j, \alpha_s^\star \rangle - \langle \lambda_j, \alpha_r^\star \rangle \langle \alpha_r, \alpha_s^\star \rangle$$

Now $\langle \lambda_j, \alpha_s^\star \rangle = 0$ since $s \in J_i \subseteq J_{j+1}$; $\langle \lambda_j, \alpha_r^\star \rangle \geq 0$ by induction; and $\langle \alpha_r, \alpha_s^\star \rangle \leq 0$. Hence $\langle \lambda_i, \alpha_s^\star \rangle \geq 0$ and property (2) is satisfied.

Finally note that since the subsets J_i are finite and each is strictly contained in the last, we eventually get $J_{k+1} = \emptyset$. Hence $W_{\lambda_1,\ldots,\lambda_k} = W_{J_{k+1}} = 1$ and so the weights $\lambda_1, \ldots, \lambda_k$ are a base for W. □

The weights λ_i and sets J_i can be constructed using Algorithm 3. Suppose that w is an element of the Weyl group W. Then we have a unique factorisation $w = w_k \cdots w_2 w_1$ where w_i is the unique $W_{J_{i+1}}$-reduced element of W_{J_i} such that $\lambda_i w_i = \lambda_i w w_1^{-1} \cdots w_{i-1}^{-1}$. The element w_{0i} is fixed for a particular group G, and is the unique $W_{J_{i+1}}$-reduced element of W_{J_i} such that $\lambda_i w_{0i}^{-1} = \lambda_i w_{0J_i}$. Iterating the result in the previous subsection, we can now refine the Bruhat decomposition to

$$\bigcup_{w_1,\ldots,w_k} U_{w_{01}} U_{w_{02}} \cdots U_{w_{0k}} H \dot{w}_k \cdots \dot{w}_2 \dot{w}_1 (U_{w_k})^{\dot{w}_{k-1} \cdots \dot{w}_1} \cdots (U_{w_2})^{\dot{w}_1} U_{w_1}.$$

where w_j ranges over the $W^{(j+1)}$-reduced elements of $W^{(j)}$.

7.5. **Row and column reduction.** Putting together the ideas of the previous sections, we get Algorithm 4 (see Subsection 7.7 for the significance of the asterisks). We assume that λ_i, J_i and w_{0i} have already been computed using the techniques of Subsection 7.4. Note that the loop has an invariant $\rho(u) A \rho(u')$, which is always

let $u, h, u' = 1$.
for i from 1 to k do
  [Compute w_i]
  find the largest $\mu \in \Omega$ so that $v_i A \cdot v_\mu \neq 0$.
  find the $W_{J_{i+1}}$-reduced element $w_i \in W_{J_i}$ with $\lambda_i w_i = \mu w_1^{-1} \cdots w_{i-1}^{-1}$ (*)
  [Compute the component in $(U_{w_i})^{\dot{w}_{i-1}\cdots\dot{w}_1}$]
  for β in $\Phi_{w_i w_{i-1}\cdots w_1}$ in additive order do
    let $m = \langle \mu, \beta^\star \rangle$, $c = (-1)^m (v_{\lambda_i} A \cdot v_\mu n_\beta)(v_{\lambda_i} A \cdot v_\mu)^{-1}$
    let b be the mth root of c so that $v_{\lambda_i} A x_\beta(b)^{-1} \cdot v_\mu n_\beta = 0$ (*)
    let $u' = x_\beta(b)u'$, $A = A\rho(x_\beta(b))^{-1}$
  end for.
  [Compute the component in $U_{w_{0i}}$]
  for α in $\Phi_{\dot{w}_{0i}}$ in additive order do
    let $m = \langle \lambda_i, \alpha^\star \rangle$, $c = (\varepsilon_\alpha)^m (v_{\lambda_i} n_\alpha \cdot A v_\mu)(v_{\lambda_i} \cdot A v_\mu)^{-1}$
    let a be the mth root of c so that $v_{\lambda_i} n_\alpha x_\alpha(a)^{-1} \cdot A v_\mu = 0$ (*)
    let $u = u x_\alpha(a)$, $A = \rho(x_\alpha(a))^{-1} A$
  end for
end for
let $w = w_k \cdots w_1$, $A = A\rho(\dot{w})^{-1}$

ALGORITHM 4. Generalised row and column reduction

equal to the original value of A. At the end of this algorithm, A is the image of a torus element.

7.6. **Torus elements.** We now suppose we have an element $A \in \rho(H)$ where H is the torus of our group of Lie type G, and we wish to find $h \in H$ so that $\rho(h) = A$. Note that ρ need not be faithful on H, so h may not be uniquely determined. Let Γ be the lattice generated by the weights of V and choose a basis $\gamma_1, \ldots, \gamma_t$ for Γ. Then the preimage of A in $\Gamma \otimes \mathbb{F}^\times$ is computed as $\prod_{i=1}^t \gamma_i \otimes t_i$ where $vA = t_i v$ for $v \in V_{\gamma_i}$. This can be converted to the form $\prod_{i=1}^t f_i \otimes h_i$ using the method of Subsection 5.2.

7.7. **Application to matrix group recognition.** One of the more difficult problems in matrix group recognition is finding a membership test for a known almost simple group. Our algorithm provides such a test for natural characteristic representations of finite groups of Lie type. We obtain a membership test for $\rho(G)$ by running our algorithm for an arbitrary $A \in \mathrm{GL}(V)$: A is not a member of $\rho(G)$ if the algorithm fails or the value of A at the end is not the identity matrix. Algorithm 4 can fail in three places, which are marked by (*). The torus element algorithm of Subsection 7.6 can also fail.

In this and other applications, it is of interest to determine whether our algorithm runs in polynomial time. When $\mathbb{F}$ is a finite field of order q, it is straightforward to show that most of the algorithm runs in polynomial time in $\log(q)$ and the coefficients of λ. For example, the length k of our base is bounded by the reductive rank of G and radicals over finite fields can be computed in polynomial time by Flajolet et al. (2001). The main sticking point is the computation of discrete logarithms in Subsection 7.1. This can be avoided, however, since the powers

involved are bounded by m, and by Subsection 7.3 the values of m which appear are polynomial in the coefficients of λ. We now have the following result.

Theorem 7.4. *Let $G = G_{\mathcal{R}}(\mathbb{F}_q)$ be a finite group of Lie type and let $\rho : G \to \mathrm{GL}(V)$ be a quotient of a highest weight representation with known highest weight λ. Suppose $A \in \mathrm{GL}(V)$. We can decide whether A is in $\rho(G)$ and, if it is, find a preimage in polynomial time in $\log(q)$ and the coefficients of λ.*

Note that the condition that the highest weight be known is required, since computing the highest weight requires discrete logarithms. In fact, if G is a one-dimensional torus with representation V, then computing the weight of V is equivalent to the discrete logarithm problem.

Note that this immediately gives a polynomial time algorithm for multiplying elements in G: simply convert them to matrices in the smallest degree highest weight representation, multiply the matrices, then convert back. In practice, however, this seems to be much slower than the algorithm given in Section 5.

7.8. **Example: $\mathrm{GL}_{n+1}(\mathbb{F})$.** We now describe how this algorithm works in the case of the standard representation of the general linear group. Take $X = \mathbb{Z}^{n+1}$ with standard basis vectors e_i and $Y = \mathbb{Z}^{n+1}$ with standard basis vectors f_i. The roots and coroots are of the form $\alpha_{ij} = e_i - e_j$ and $\alpha_{ij}^\star = f_i - f_j$ for $1 \le i, j \le n+1$ with $i \ne j$. A (co)root is positive when $i > j$ and simple when $i = j + 1$.

The standard representation $\rho : G \to \mathrm{GL}_{n+1}(\mathbb{F})$ takes $x_{\alpha_{ij}}(a)$ to the matrix equal to the identity except for the (i, j)-entry which is equal to a, and $f_i \otimes t$ to the matrix equal to the identity except for the (i, i)-entry which is equal to t. The Weyl group of G is the symmetric group on $n + 1$ letters with reflections $s_{\alpha_{ij}} = (i, j)$. We choose δ so that $\rho(g^\delta) = \rho(g)^t$.

Algorithm 3 gives $\lambda_i = e_i$ and $J_i = \{i, i + 1, \ldots, n + 1\}$. Since we can take $v_{\lambda_i} = b_i$, these weights correspond to the rows and columns of the input matrix A. We can compute

$$w_{0i} = s_n \ldots s_{i+1} s_i = (i, i + 1, \ldots, n + 1) \quad \text{and}$$
$$\Phi_{w_{0i}} = \{\alpha_{i+1,i}, \ldots, \alpha_{n+1,i}\}.$$

In order to compute w_i we need to know the image

$$\lambda_i w_i = \mu w_1^{-1} \cdots w_{i-1}^{-1}$$

where $\mu = e_j$ with j is the largest nonzero entry in the ith row of A. Now

$$w_i = s_i s_{i+1} \ldots s_{iw_i - 1} = (iw_i, \ldots, i + 1, i) \quad \text{and}$$
$$\Phi_{w_i} = \{\alpha_{iw_i,i},\ \alpha_{iw_i,i+1},\ \ldots, \alpha_{iw_i,iw_i-1}\}$$

and so

$$\Phi_{w_i} w_{i-1} \cdots w_1 = \{\alpha_{iw,j} \mid j = l w_{i-1} \cdots w_1 \text{ for } l = i, i + 1, \ldots iw_i - 1\} \tag{24}$$

Furthermore the constants m that appear in Algorithm 4 are all equal to 1. It is now easily seen that computing the component in $U_{w_{0i}}$ is just clearing the $(iw_{i-1} \cdots \dot{w}_1)$th column of the matrix with row operations, and computing the component in $(U_{w_i})^{\dot{w}_{i-1} \cdots \dot{w}_1}$ is just clearing the ith row of the matrix with column operations (the values of j that appear in (24) give exactly the entries in the row that have not been cleared by a previous column operation).

In order to relate this algorithm to the well-known LUP algorithm, we note that ρ can be chosen so that $\rho(B)$ consists of lower triangular matrices. Suppose that

$g \in G$ decomposes as $uh\dot{w}u'$ for $u \in U$, $h \in H$, $w \in W$ and $u' \in U_w$. Then $L = \rho(uh)$ is lower triangular, $U = \rho(\dot{w}u'\dot{w}^{-1})$ is upper triangular, and $P = \rho(\dot{w})$ is a permutation matrix (up to signs), giving the usual LUP decomposition.

References

J. L. Alperin and Rowen B. Bell. *Groups and representations*. Springer-Verlag, New York, 1995.

L. Babai, A. J. Goodman, W. M. Kantor, E. M. Luks, and P. P. Pálfy. Short presentations for finite groups. *J. Algebra*, 194(1):79–112, 1997.

W. W. Bosma and J. J. Cannon. *Handbook of Magma functions*. School of Mathematics and Statistics, University of Sydney, Sydney, 1997.

Nicolas Bourbaki. *Éléments de mathématique*. Masson, Paris, 1981. Groupes et algèbres de Lie. Chapitres 4, 5 et 6. [Lie groups and Lie algebras. Chapters 4, 5 and 6].

G. Butler. *Fundamental algorithms for permutation groups*. Springer-Verlag, Berlin, 1991.

R. W. Carter. Simple groups and simple Lie algebras. *J. London Math. Soc.*, 40: 193–240, 1965.

Roger W. Carter. *Simple groups of Lie type*. John Wiley & Sons, London-New York-Sydney, 1972. Pure and Applied Mathematics, Vol. 28.

Roger W. Carter. *Finite groups of Lie type*. John Wiley & Sons Ltd., Chichester, 1993. Conjugacy classes and complex characters, Reprint of the 1985 original, A Wiley-Interscience Publication.

W. A. de Graaf. Constructing representations of split semisimple Lie algebras. *J. Pure Appl. Algebra*, 164(1-2):87–107, 2001. Effective methods in algebraic geometry (Bath, 2000).

M. Demazure. Données radicielles. In *Schémas en Groupes (Sém. Géométrie Algébrique, Inst. Hautes Études Sci., 1964), Fasc. 6, Exposé 21*. Inst. Hautes Études Sci., Paris, 1965.

P. Flajolet, X. Gourdon, and D. Panario. The complete analysis of a polynomial factorization algorithm over finite fields. *J. Algorithms*, 40(1):37–81, 2001.

I. B. Frenkel and V. G. Kac. Basic representations of affine Lie algebras and dual resonance models. *Invent. Math.*, 62(1):23–66, 1980/81.

The GAP Group, Aachen, St Andrews. *GAP – Groups, Algorithms, and Programming, Version 4.1*, 1999. (`http://www-gap.dcs.st-and.ac.uk/~gap`).

Meinolf Geck, Gerhard Hiss, Frank Lübeck, Gunter Malle, and Götz Pfeiffer. CHEVIE—a system for computing and processing generic character tables. *Appl. Algebra Engrg. Comm. Comput.*, 7(3):175–210, 1996. Computational methods in Lie theory (Essen, 1994).

Peter B. Gilkey and Gary M. Seitz. Some representations of exceptional Lie algebras. *Geom. Dedicata*, 25(1-3):407–416, 1988. Geometries and groups (Noordwijkerhout, 1986).

Sergei Haller. Unipot—a system for computing with elements of unipotent subgroups of chevalley groups, version 1.1. Technical report, Justus-Liebig Universitaet, Germany, July 2000. `ftp://ftp-pslabor.hrz.uni-giessen.de/SHadow/unipot/`.

R. B. Howlett, L. J. Rylands, and D. E. Taylor. Matrix genrators for the exceptional groups of lie type. Report 99-14, School of Mathematics and Statistics, The University of Sydney, June 1999.

Alexander Hulpke and Ákos Seress. Short presentations for three-dimensional unitary groups. *J. Algebra*, 245(2):719–729, 2001.

William M. Kantor and Ákos Seress. Black box classical groups. *Mem. Amer. Math. Soc.*, 149(708):viii+168, 2001.

C. R. Leedham-Green and L. H. Soicher. Collection from the left and other strategies. *J. Symbolic Comput.*, 9(5-6):665–675, 1990. Computational group theory, Part 1.

Charles R. Leedham-Green. The computational matrix group project. In *Groups and computation, III (Columbus, OH, 1999)*, pages 229–247. de Gruyter, Berlin, 2001.

Martin W. Liebeck and Jan Saxl. On the orders of maximal subgroups of the finite exceptional groups of Lie type. *Proc. London Math. Soc. (3)*, 55(2):299–330, 1987.

R. J. Riebeek. *Computations in association schemes*. PhD thesis, Eindhoven University of Technology, 1998.

L. J. Rylands. A formula for signs of structure constants. Unpublished, 2001.

Charles C. Sims. *Computation with finitely presented groups*. Cambridge University Press, Cambridge, 1994.

T. A. Springer. *Linear algebraic groups*. Birkhäuser Boston Inc., Boston, MA, second edition, 1998.

R. Steinberg. Lectures on chevalley groups. Technical report, Yale University, 1968.

Robert Steinberg. Générateurs, relations et revêtements de groupes algébriques. In *Colloq. Théorie des Groupes Algébriques (Bruxelles, 1962)*, pages 113–127. Librairie Universitaire, Louvain, 1962.

Robert Steinberg. Representations of algebraic groups. *Nagoya Math. J.*, 22:33–56, 1963.

M.A.A. van Leeuwen, A.M. Cohen, and B. Lisser. *LiE Manual*. CWI/CAN, Amsterdam, 1992. Manual for the software package LiE for Lie group theoretical computations. `http://thales.sp2mi.univ-poitiers.fr /~maavl/LiE`.

Arjeh M. Cohen & Scott Murray, Department of Mathematics and Computer Science, Eindhoven University of Technology, PO Box 513, 5600 MB Eindhoven, The Netherlands

E-mail address: `A.M.Cohen@tue.nl, smurray@win.tue.nl`

Don E. Taylor, Department of Mathematics and Statistics, University of Sydney, Sydney, NSW 2006, Australia

E-mail address: `don@maths.usyd.edu.au`