Constructive recognition of classical groups in odd characteristic

C.R. Leedham-Green and E.A. O'Brien

Abstract

Let $G = \langle X \rangle \leq \mathrm{GL}(d, F)$ be one of $\mathrm{SL}(d, F)$, $\mathrm{Sp}(d, F)$, or $\mathrm{SU}(d, F)$ where F is a finite field of odd characteristic. We present recognition algorithms to construct standard generators for G which allow us to write an element of G as a straight-line program in X. The algorithms are Las Vegas polynomial-time, subject to the existence of a discrete log oracle for F.

1 Introduction

A major goal of the "matrix recognition project" is the development of efficient algorithms for the investigation of subgroups of $\mathrm{GL}(d, F)$ where F is a finite field. We refer to the recent survey by O'Brien [25] for background related to this work. A particular aim is to identify the composition factors of $G \leq \mathrm{GL}(d, F)$. If a problem can be solved for the composition factors, then it can be frequently be solved for G.

One may intuitively think of a *straight-line program* (SLP) for $g \in G = \langle X \rangle$ as an efficiently stored group word on X that evaluates to g. For a formal definition, we refer the reader to Seress [28, p. 10]. WE SHOULD GIVE THE DEFINITION, E.G. IN SECTION 11 A critical property of an SLP is that its length is proportional to the number of multiplications and exponentiations used in constructing the corresponding group element.

A *constructive recognition algorithm* constructs an explicit isomorphism between a group G and a "standard" (or natural) representation of G, and exploits this isomorphism to write an arbitrary element of G as an SLP in its defining generators.

In this paper we present constructive recognition algorithms for certain of the classical groups. Let $\mathrm{SX}(d, q)$ denote $\mathrm{SL}(d, q)$, or $\mathrm{Sp}(d, q)$ (for even d), or $\mathrm{SU}(d, q^2)$, and let $\mathrm{GX}(d, q)$ denote $\mathrm{GL}(d, q)$, or $\mathrm{Sp}(d, q)$, or $\mathrm{U}(d, q^2)$. We present and analyse two algorithms that take as input a generating subset X of $\mathrm{SX}(d, q)$ for q odd, and return as output *standard generators* of this group as SLPs in X. Usually, these generators

This work was supported in part by the Marsden Fund of New Zealand via grant UOA 412. 2000 *Mathematics Subject Classification.* Primary 20C20, 20C40.

are defined with respect to a basis different to that for which X was defined, and a change-of-basis matrix is also returned to relate these bases.

Similar algorithms are under development for the orthogonal groups in odd characteristic. Further, characteristic 2 can also be addressed in the same style, but the resulting algorithms are more complex. We shall consider these cases in later papers.

Our principal result is the following.

Theorem 1.1 *Let $G = \langle X \rangle \leq \mathrm{GL}(d,q)$ denote $\mathrm{SL}(d,q)$, or $\mathrm{Sp}(d,q)$ for even d, or $\mathrm{SU}(d,q^2)$ where, in all cases, q is odd. There are Las Vegas algorithms which, given the input X, construct a new standard generating set S for G having the property that an SLP of length $O(d^2 \log q)$ can be found from S for any $g \in G$. Assuming the existence of a discrete log oracle for $\mathrm{GF}(q)$, the algorithm to construct S runs in $O(d(\xi + d^3 \log q + \chi))$ field operations, where ξ is the cost of constructing an independent (nearly) uniformly distributed random element of G, and χ is the cost of a call to a discrete log oracle for $\mathrm{GF}(q)$.*

We prove this theorem by exhibiting algorithms with the stated complexity. If we assume that a random element can be constructed in $O(d^3)$ field operations, then $O(d^4)$ is an upper bound to the complexity for fixed q.

Brooksbank's algorithms [5] for the natural representation of $\mathrm{Sp}(d,q)$, $\mathrm{SU}(d,q)$, and $\Omega^\epsilon(d,q)$ have complexity $O(d^5)$ for fixed q. More precisely, the complexity of his algorithm is

$$O(d^3 \log q(d + \log d \log^3 q) + \xi(d + \log\log q) + d^5 \log^2 q + \chi(\log q)).$$

The algorithm of Celler & Leedham-Green [10] for $\mathrm{SL}(d,q)$ has complexity $O(d^4 \cdot q)$.

The two algorithms presented here reflect a tension between two competing tasks: the speed of construction of the standard generators, and minimising the length of the resulting SLPs for the standard generators in X. The first is designed for optimal efficiency; the second to produce short SLPs.

We establish some notation. Let $g \in G \leq \mathrm{GL}(d,q)$, let $\bar{G}$ denote $G/G \cap Z$ where Z denotes the centre of $\mathrm{GL}(d,q)$, and let $\bar{g}$ denote the image of g in $\bar{G}$. The *projective centraliser* of $g \in G$ is the preimage in G of $C_{\bar{G}}(\bar{g})$. Further $g \in G$ is a *projective involution* if g^2 is scalar, but g is not.

A central component of both algorithms is the use of involution centralisers. In Section 2 we summarise the structure of involution centralisers for elements of classical groups in odd characteristic. In Section 3 we define standard generators for the classical groups. A version of the "generalised echelon" algorithm of Cohen, Murray & Taylor [11] can be used to write a given element of a classical group in terms of these generators.

In Sections 4 and 5 the two algorithms are described. They rely on finding involutions whose eigenspaces have approximately the same dimension in the case of the first algorithm, and exactly the same dimension in the second. The construction of such involutions is described and analysed in Section 6. The centraliser of an involution is constructed using an algorithm of Bray [4]; this is considered in Section 7. The base

cases of the algorithms (when $d \leq 4$) are discussed in Section 8. We frequently compute high powers of elements of linear groups; an algorithm for doing this efficiently is described in Section 9. The use of powering to construct the direct factors from the direct product of two classical groups is discussed in Section 10. The complexity of the algorithms and the length of the resulting SLPs for the standard generators are discussed in Section 11. Finally we report on our implementation of the algorithm, publicly available in Magma [6].

2 Centralisers of involutions in classical groups

We now briefly review the structure of involution centralisers in (projective) classical groups defined over fields of odd characteristic. A detailed account can be found in [14].

1. If u is an involution in $\mathrm{SL}(d,q)$, with eigenspaces E_+ and E_-, then the centraliser of u in $\mathrm{SL}(d,q)$ is $(\mathrm{GL}(E_+) \times \mathrm{GL}(E_-)) \cap \mathrm{SL}(d,q)$. The centraliser of the image of u in $\mathrm{PSL}(d,q)$ is the image of the centraliser of u in $\mathrm{SL}(d,q)$ if E_+ and E_- have different dimensions. If E_+ and E_- have the same dimension, then in $\mathrm{PSL}(d,q)$ these eigenspaces may be interchanged by the centraliser of the image of u, which is now the image of $(\mathrm{GL}(d/2,q) \wr C_2) \cap \mathrm{SL}(d,q)$ in $\mathrm{PSL}(d,q)$.

2. If u is an involution in $\mathrm{Sp}(2n,q)$, with eigenspaces E_+ and E_-, these spaces are mutually orthogonal, and the form restricted to either is non-singular. Thus the centraliser of u is $\mathrm{Sp}(E_+) \times \mathrm{Sp}(E_-)$. The centraliser of the image of u in $\mathrm{PSp}(2n,q)$ is the image of $\mathrm{Sp}(E_+) \times \mathrm{Sp}(E_-)$, except when the eigenspaces have the same dimension, when the centraliser again permutes the eigenspaces. An element of the projective centraliser permuting the eigenspaces sends (v,w) to $(w\theta, -v\theta)$, where θ is an isometry that permutes these spaces, so the image of $\mathrm{Sp}(E_+) \times \mathrm{Sp}(E_-)$ has index 2 in the projective centraliser.

3. If u is an involution in $\mathrm{SU}(d,q^2)$, the situation is similar. Again the eigenspaces of u are mutually orthogonal, and the form restricted to the eigenspaces is non-degenerate. The centraliser of u in $\mathrm{SU}(d,q^2)$ is $(\mathrm{U}(E_+) \times \mathrm{U}(E_-)) \cap \mathrm{SU}(d,q^2)$. The centraliser of the image of u in $\mathrm{PSU}(d,q^2)$ is the image of the centraliser of u in $\mathrm{SU}(d,q^2)$ except where the eigenspaces of u have the same dimension, when the centraliser is the image of $(\mathrm{U}(d/2,q^2) \wr C_2) \cap \mathrm{SU}(d,q^2)$ in $\mathrm{PSU}(d,q^2)$.

3 Standard generators for classical groups

We now describe *standard generators* for the perfect classical groups $\mathrm{SL}(d,q)$, $\mathrm{Sp}(d,q)$ and $\mathrm{SU}(d,q^2)$ where q is odd in all cases.

We use the notation $\mathrm{SX}(d,q)$ to denote any one of these groups, and $\mathrm{PX}(d,q)$ to denote the corresponding central quotient.

Let V be the natural module for a perfect classical group G of the above kind. We define a *hyperbolic* basis for V as follows. If $G = \mathrm{SL}(d, q)$ then any ordered basis is hyperbolic. If $G = \mathrm{Sp}(d, q)$ then d is even, say $d = 2n$, and G preserves a non-degenerate symplectic form. A hyperbolic basis for V is then an ordered basis of the form $\{e_1, f_1, \ldots, e_n, f_n\}$, where, if the image of a pair of vectors (v, w) under the form is written as $v.w$, then $e_i.e_j = f_i.f_j = 0$ for all i, j (including the case $i = j$), and $e_i.f_j = 0$ for $i \neq j$, and $e_i.f_i = -f_i.e_i = 1$ for all i. If $G = \mathrm{SU}(d, q^2)$, and $d = 2n$ is even, then the definition is exactly as for the case of $\mathrm{Sp}(d, q)$ except that, the form being hermitian, the condition $e_i.f_i = -f_i.e_i = 1$ for all i is replaced by the condition $e_i.f_i = f_i.e_i = 1$ for all i. If $G = \mathrm{SU}(d, q^2)$, where $d = 2n + 1$, a hyperbolic basis is of the form $(e_1, f_1, \ldots, e_n, f_n, v)$, where the above equations hold, and in addition $e_i.v = f_i.v = 0$ for all i, and $v.v = 1$.

That a hyperbolic basis exists for V is easily established; it can be constructed from an arbitrary basis in $O(d^3)$ field operations. For details, see for example, [15, Chapter 2].

The standard generators introduced here are defined in terms of a hyperbolic basis for V, which will be defined in terms of the given basis by a change-of-basis matrix.

It is of course a triviality to *write down* the standard generators (once they have been defined). However we must construct these elements as SLPs in the given generators.

Once a hyperbolic basis has been chosen for V, the Weyl group of G can be defined as a section of G, namely as the group of monomial matrices in G modulo diagonal matrices, thus defining a subgroup of the symmetric group S_d. For $G = \mathrm{SL}(d, q)$, this group is S_d. For $\mathrm{Sp}(2n, q)$ the Weyl group is the subgroup of S_{2n} that preserves the system of imprimitivity with blocks $\{e_i, f_i\}$ for $1 \leq i \leq n$, and is thus $C_2 \wr S_n$. For each of $\mathrm{SU}(2n, q^2)$ and $\mathrm{SU}(2n + 1, q^2)$, the Weyl group is also $C_2 \wr S_n$.

In detail, the standard generating set Y for G with respect to a hyperbolic basis for V is as follows:

1. If $G = \mathrm{SL}(d, q)$ then $Y = \{s, \delta, u, v\}$ is defined as follows. All but v lie in the copy of $\mathrm{SL}(2, q)$ that normalises $\langle e_1, e_2 \rangle$ and centralises $\langle e_3, \ldots, e_d \rangle$, and these act on $\langle e_1, e_2 \rangle$ with respect to this ordered basis as follows:

$$s = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \delta = \begin{pmatrix} \omega & 0 \\ 0 & \omega^{-1} \end{pmatrix} \quad u = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

   where ω is a primitive element for $\mathrm{GF}(q)$. Finally v is defined by $e_1 \mapsto e_d \mapsto -e_{d-1} \mapsto -e_{d-2} \mapsto -e_{d-3} \cdots \mapsto -e_1$, the signs chosen to ensure that v has determinant 1. Clearly u and v generate the Weyl group, modulo the group of diagonal matrices. Note that $v = u$ if $d = 2$.

2. If $G = \mathrm{Sp}(d, q)$, where $d = 2n$ and $n > 1$, then $Y = \{s, t, \delta, u, v\}$ where s and δ are as defined for $\mathrm{SL}(d, q)$; and t is the element of G that centralises $\langle e_i, f_i : i > 2 \rangle$, normalises the space $\langle e_1, f_1, e_2, f_2 \rangle$, and acts on the space with matrix referred to

this hyperbolic basis given by

$$t = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix};$$

and u and v are permutation matrices defined by $u = (e_1, e_2)(f_1, f_2)$ and $v = (e_1, e_2, \ldots, e_n)(f_1, f_2, \ldots, f_n)$. Note that $v = u$ if $n = 2$.

3. If $G = \mathrm{SU}(d, q^2)$, where $d = 2n$ and $n > 1$, then $Y = \{s, t, \delta, u, v\}$, where u and v are as defined for $\mathrm{Sp}(d, q)$; and δ and s both centralise all but the first two basis vectors, normalise the space spanned by the first two basis vectors, and act on this space, with respect to the ordered basis (e_1, f_1) as

$$\delta = \begin{pmatrix} \omega & 0 \\ 0 & \omega^{-q} \end{pmatrix} \quad s = \begin{pmatrix} 1 & \alpha \\ 0 & 1 \end{pmatrix}$$

where ω is a primitive element of $\mathrm{GF}(q^2)$, and $\alpha = \omega^{(q+1)/2}$; and t centralises all but the first four basis vectors, normalises the space spanned by the first four basis elements, and acts on this space, with respect to the ordered basis (e_1, f_1, e_2, f_2) as

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

4. If $G = \mathrm{SU}(d, q^2)$, where $d = 2n + 1$ and $n > 0$, then $Y = \{s, t, x, y, \delta, u, v\}$, where the generators except for x and y are as for the even case, but with t omitted if $d = 3$. Now x and y centralise all but the first two and the last basis vectors, normalise the space that these three vectors span, and act on this space with respect to the ordered basis (e_1, f_1, v) with matrices of the form

$$\begin{pmatrix} 1 & \beta & \gamma \\ 0 & 1 & -\beta^q \\ 0 & 0 & 1 \end{pmatrix}.$$

In each case the equation $\beta^{q+1} + \gamma + \gamma^q = 0$ is satisfied. The two values of β (one for x and one for y) are chosen so that they span $\mathrm{GF}(q^2)$ over $\mathrm{GF}(q)$.

In all cases, these generators have the property that it is easy to construct from them any element of any root group, and consequently these generators do generate the group in question. The root groups are defined with respect to a maximal split torus, which we take to be the group of diagonal matrices in the group in question (with the additional restriction, in the case of $\mathrm{SU}(2n + 1, q^2)$, that the final diagonal entry is 1). These root groups can then be constructed as follows.

1. If $G = \mathrm{SL}(d, q)$ then the root groups are of the form $\{s^{\delta^i g} : 0 \leq i \leq q-2\}$, where $g \in \langle u, v \rangle$.

2. If $G = \mathrm{Sp}(2n, q)$ then the root groups corresponding to short roots are again of the form $\{s^{\delta^i g} : 0 \leq i \leq q-2\}$, where $g \in \langle u, v \rangle$, and root groups corresponding to long roots are of the form $\{t^{\delta^i g} : 0 \leq i \leq q-2\}$, where $g \in \langle u, v \rangle$.

3. If $G = \mathrm{SU}(2n, q)$ then the root groups are defined by the same formulae as in the case $G = \mathrm{Sp}(2n, q)$.

4. If $G = \mathrm{SU}(2n+1, q)$ then the root groups are as in the previous case, together with a family of two-parameter groups, namely the set of elements that normalise the space spanned by $\{e_i, f_i, v\}$, centralise the other basis elements, and act on the above 3-space, with respect to the ordered basis (e_i, v, f_i), as the set of matrices of the form
$$\begin{pmatrix} 1 & \beta & \gamma \\ 0 & 1 & -\beta^q \\ 0 & 0 & 1 \end{pmatrix},$$
where the equation $\beta^{q+1} + \gamma + \gamma^q = 0$ is satisfied. This is a non-abelian group of order q^3. Its derived group and centre coincide, and these form the set of matrices with $\beta = 0$. If $i = 1$ then conjugating by δ multiplies all three matrix entries above the diagonal by a primitive element of $\mathrm{GF}(q)$, so these root groups can be written as $\{(x^{\delta^i} y^{\delta^j} [x, y]^{\delta^k})^g : 0 \leq i, j, k \leq q-2\}$ for $g \in \langle u, v \rangle$.

Define $Y_0 := \{s, \delta, u, v\}$. If G is $\mathrm{Sp}(2n, q)$ where $n > 1$, or $\mathrm{SU}(2n, q^2)$ or $\mathrm{SU}(2n+1, q^2)$ for $n > 0$, then Y_0 generates $\mathrm{SL}(2, q) \wr S_n$. For these groups, the first and major step in our algorithm constructs Y_0. As a final step, we construct the additional element or elements to obtain Y.

If $G = \mathrm{SL}(2n, q)$, the first step also constructs $\mathrm{SL}(2, q) \wr S_n$; in a final step we obtain the $2n$-cycle.

4 Algorithm `One`

Algorithm `One` takes as input a generating set X for $G = \mathrm{SX}(d, q)$, and returns standard generators for G as SLPs in X. The generators are in standard form when referred to a basis constructed by the algorithm. The change-of-basis matrix that expresses this basis in terms of the standard basis for the natural module is also returned.

The algorithm employs a "divide-and-conquer" strategy. Define a *strong involution* in $\mathrm{SX}(d, q)$ to be an involution whose eigenspaces have dimensions in the range $(d/3, 2d/3]$ if $d > 5$, and in the range $[2, 3]$ if $d = 5$. For $\mathrm{Sp}(d, q)$ and $\mathrm{SU}(d, q^2)$ the eigenspaces of an involution u are mutually orthogonal, and the form restricted to either eigenspace is non-degenerate. Thus, if these spaces have dimensions e and $d - e$, then the derived subgroup of the centraliser of u in $\mathrm{SX}(d, q)$ is $\mathrm{SX}(e, q) \times \mathrm{SX}(d-e, q)$.

Note that the dimension of the -1-eigenspace of an involution in $\mathrm{SX}(d,q)$ is always even.

Algorithm `OneEven` addresses the case of even d.

Algorithm 1: `OneEven`$(X, type)$

```
/* X is a generating set for the perfect classical group G in odd
   characteristic, of type SL or Sp or SU, in even dimension.
   Return standard generating set Y_0 for a copy of SL(2,q) ≀ S_{d/2} ≤ G,
   the SLPs for the elements of Y_0, the change-of-basis matrix, and
   generators for centraliser of involution k defined in line 13.
   */
```

1 **begin**
2 $d :=$ the rank of the matrices in X;
3 if $d \leq 4$ then return `BaseCase` (X, *type, false*);
4 $q :=$ the size of the field over which these matrices are defined;
5 if *type* $=$ SU then $q := q^{1/2}$;
6 Find by random search $g \in G := \langle X \rangle$ of even order such that g powers to a strong involution h;
7 Let n be the dimension of the $+1$-eigenspace of h;
8 Find generators for the centraliser C of h in G;
9 In the derived subgroup C' of C find generating sets X_1 and X_2 for the direct factors of C';
10 $(s_1, \delta_1, u_1, v_1) :=$ `OneEven`$(X_1, type)$;
11 $(s_2, \delta_2, u_2, v_2) :=$ `OneEven`$(X_2, type)$;
12 Let $(e_1, f_1, \ldots, e_n, f_n, e_{n+1}, f_{n+1}, \ldots, e_d, f_d)$ be the concatenation of the hyperbolic bases constructed in lines 10 and 11;
13 $k := (\delta_1^{(q-1)/2})^{v_1^{-1}} \delta_2^{(q-1)/2}$;
14 Find generators for the centraliser D of k in G;
15 In the derived subgroup D' of D find a generating set X_3 for the direct factor that acts faithfully on $\langle e_n, e_{n+1}, f_n, f_{n+1} \rangle$;
16 In $\langle X_3 \rangle$ find the permutation matrix $j = (e_n, e_{n+1})(f_n, f_{n+1})$;
17 $v := v_1 j v_2$;
18 return (s_1, δ_1, u_1, v), the change-of-basis matrix, and X_3.
19 **end**

If the type is SL, then the centraliser of h is $\mathrm{GL}(E_+) \times \mathrm{GL}(E_-) \cap \mathrm{SL}(d,q)$ where E_+ and E_- are the eigenspaces of h. If the type is Sp, it is $\mathrm{Sp}(E_+) \times \mathrm{Sp}(E_-)$, and if the type is SU, it is $(U(E_+) \times U(E_-)) \cap \mathrm{SU}(d,q^2)$. In these last two cases the restriction of the form to each of the eigenspaces is non-singular, and each eigenspace is orthogonal to the other. Thus the concatenation of a hyperbolic basis of one eigenspace with a hyperbolic basis for the other eigenspace is a hyperbolic basis for the whole space.

As presented the algorithm has been simplified. In lines 11 and 12 we have ignored the change-of-basis matrices that are also returned; the change-of-basis returned at line

18 is the concatenation of these bases.

We make the following additional observations on Algorithm `OneEven`.

1. The SLPs that express the standard generators in terms of X are also returned.

2. Generators for the involution centraliser in line 8 are constructed using the algorithm of Bray [4], see Section 7. Of course, g is an element of this centraliser. We need only a subgroup of the centraliser that contains its derived subgroup.

3. The generators for the direct summands constructed in line 9 are constructed by forming suitable powers of the generators of the centraliser. This step is discussed in Section 10.

4. The algorithms for the `BaseCase` calls in lines 3 and 16 are discussed in Section 8.

5. The search for an element that powers to a suitable involution is discussed in Section 6.

6. The recursive calls in lines 10 and 11 are in smaller dimension. Not only are the groups of smaller Lie rank, but the matrices have degree at most $2d/3$. Hence these calls only affect the time or space complexity of the algorithm up to a constant multiple; however they contribute to the length of the SLPs produced.

7. Note that k in line 12 is an involution: its -1-eigenspace is $\langle e_n, f_n, e_{n+1}, f_{n+1}\rangle$ and its $+1$-eigenspace is $\langle e_1, f_1, \ldots, e_{n-1}, f_{n-1}, e_{n+2}, f_{n+2}, \ldots, e_d, f_d\rangle$.

Algorithm `OneOdd`, which considers the case of odd degree d, is similar to Algorithm `OneEven`. Our commentary on the even degree case also applies.

Algorithm 2: `OneOdd`$(X, type)$

```
/* X is a generating set for the perfect classical group G in odd
   characteristic and degree, of type SL or SU. If G = SL(d, q),
   then return standard generating set Y for G; if G = SU(d, q^2)
   then return generating set Y_0 for SL(2, q) ≀ S_{(d-1)/2}.  Also return
   the SLPs for elements of this generating set, the
   change-of-basis matrix, and generators for centraliser of
   involution k defined in line 13.                                  */
1  begin
2      d := the rank of the matrices in X;
3      if d = 3 then return BaseCase (X, type, false);
4      q := the size of the field over which these matrices are defined;
5      if type = SU then q := q^{1/2};
6      Find by random search g ∈ G := ⟨X⟩ of even order such that g powers to a
       strong involution h;
7      Let n be the dimension of the +1-eigenspace of h;
8      Find generators for the centraliser C of h in G;
9      In the derived subgroup C' of C find generating sets X_1 and X_2 for the
       direct factors of C', where X_1 centralises the −1-eigenspace of h;
10     (s_1, δ_1, u_1, v_1) := OneOdd(X_1, type);
11     (s_2, δ_2, u_2, v_2) := OneEven(X_2, type);
12     Let (e_1, f_1, ..., e_n, f_n, e_{n+1}, f_{n+1}, ..., e_d, f_d) be the concatenation of the
       hyperbolic bases constructed in lines 10 and 11;
13     k := (δ_1^{(q-1)/2})^{v_1^{-1}} δ_2^{(q-1)/2};
14     Find generators for the centraliser D of k in G;
15     In the derived subgroup D' of D find a generating set X_3 for the direct factor
       that acts faithfully on ⟨e_n, e_{n+1}, f_n, f_{n+1}⟩;
16     In ⟨X_3⟩ find the permutation matrix j = (e_n, e_{n+1})(f_n, f_{n+1});
17     v := v_1 j v_2;
18     return (s_1, δ_1, u_1, v), the change-of-basis matrix and X_3.
19 end
```

We summarise the main algorithm as Algorithm `OneMain`.

If $G = \mathrm{SL}(2n, q)$ and $n > 2$, we construct an additional element a which is used to construct a $2n$-cycle. It is an element of the centraliser of the involution k computed in each of `OneEven` and `OneOdd`. It acts on the subspace spanned by the basis vectors $e_n, f_n, e_{n+1}, f_{n+1}$ as follows:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \end{pmatrix}$$

and centralises the remaining $2n - 4$ basis vectors. The product av is a $2n$-cycle; we

perform a change-of-basis that permutes the basis and changes sign to produce the desired one.

Algorithm 3: `OneMain`$(X, type)$

```
/* X is a generating set for the perfect classical group G in odd
   characteristic, of type SL or Sp or SU. Return standard
   generators Y for G, the SLPs for these generators, and
   change-of-basis matrix.                                    */
1  begin
2      d := the rank of the matrices in X;
3      if d ≤ 4 then return BaseCase(X, type, true);
4      if type=SL then
5          return (s, δ, u, v) and the change-of-basis matrix;
6      end
7      (s, δ, u, v), X_3 := OneEven(X, type);
8      v := (av)^{-1};
9      change basis to obtain desired d-cycle v;
10     return (s, δ, u, v) and the change-of-basis matrix.
end
```

In $\langle X_3 \rangle$ cons

The correctness and complexity of this algorithm, and the lengths of the resulting SLPs for the standard generators, are discussed in the rest of this paper.

5 Algorithm `Two`

We present a variant of the algorithms in Section 4 based on one recursive call rather than two. Again we denote the groups $\mathrm{SL}(d,q)$, $\mathrm{Sp}(d,q)$ and $\mathrm{SU}(d,q^2)$ by $\mathrm{SX}(d,q)$, and the corresponding projective group by $\mathrm{PX}(d,q)$.

The key idea is as follows. Suppose that d is a multiple of 4. We find an involution $h \in \mathrm{SX}(d,q)$, as in line 7 of `OneEven`, but insist that it should have both eigenspaces of dimension $d/2$.

Let $\bar{h}$ be the image of h in $\mathrm{PX}(d,q)$. The centraliser of $\bar{h}$ in $\mathrm{PX}(d,q)$ acts on the pair of eigenspaces E_+ and E_- of h, interchanging them. In practice, we construct the projective centraliser of h by applying the algorithm of [4] to $\bar{h}$ and $\mathrm{PX}(d,q)$, but with the additional requirement that we find $\bar{g} \in \mathrm{PX}(d,q)$ that interchanges the two eigenspaces.

If we now find recursively a set $\mathcal{S}$ of standard generators for $\mathrm{SX}(E_+)$ with respect the basis $\mathcal{B}$, then $\mathcal{S}^g$ is a set of standard generators for $\mathrm{SX}(E_-)$ with respect to the basis $\mathcal{B}^g$. We now use these to construct standard generators for $\mathrm{SX}(d,q)$ exactly as in Algorithm `One`.

If d is an odd multiple of 2, we find an involution with one eigenspace of dimension exactly 2. The centraliser of this involution gives us $\mathrm{SX}(2,q)$ and $\mathrm{SX}(d-2,q)$. The $d-2$ factor is now processed as above, since $d-2$ is a multiple of 4, and the 2 and $d-2$ factors are combined as in the first algorithm. Thus the algorithm deals with

$\mathrm{SX}(d,q)$, for even values of d, in a way that is similar in outline to the familiar method of powering, that computes a^n, by recursion on n, as $(a^2)^{n/2}$ for even n and as $a(a^{n-1})$ for odd n.

Algorithms `TwoTimesFour` and `TwoTwiceOdd` describe the case of even d.

Algorithm `TwoTimesFour` calls no new procedures except in line 6, where we construct an involution with eigenspaces of equal dimension. This construction is discussed in Section 6.

Algorithm `TwoEven`, which summarises the even degree case, returns the generating set Y_0 defined in Section 3. We complete the construction of Y exactly as in Section 4.

If d is odd, then we find an involution whose -1-eigenspace has dimension 3, thus splitting d as $(d-3)+3$. Since $d-3$ is even, we apply the odd case precisely once.

The resulting `TwoOdd` is the same as `OneOdd`, except that it calls `TwoEven` rather than `OneEven`; similarly `TwoMain` calls `TwoOdd` and `TwoEven`.

The primary advantage of the second algorithm lies in its one recursive call. This significantly reduces the lengths of the SLPs for the standard generators.

6 Finding involutions

In the first step of our main algorithms, as outlined in Sections 4 and 5, by random search, we obtain an element of even order that has as a power a strong involution. We wish to establish a lower bound to the proportion of elements of $\mathrm{SX}(d,q)$ that power up to give a strong involution. We denote the natural module for $\mathrm{SX}(d,q)$ by V. A matrix is said to be *s*eparable if its characteristic polynomial has no repeated factors.

Lemma 6.1 *Let* $\mathrm{SX}(d,q) \leq G \leq \mathrm{GX}(q)$. *Then the proportion of separable elements of* G *is greater than*

???

Proof: [16]. □

We first estimate the probability that a random element of $\mathrm{GL}(d,q)$ has a power that is an involution having an eigenspace of dimension within a given range. Since we estimate within an error that is $O(1/q)$, we may assume, by Lemma 6.1, that the characteristic polynomial of g has no repeated factors. Thus the natural module V splits up as the direct sum of the irreducible $\langle g \rangle$-submodules of V.

Lemma 6.2 *The number of irreducible monic polynomials of degree* e *with coefficients in* $\mathrm{GF}(q)$ *is* k *where* $(q^e-1)/e > k > q^e(1-q^{-1})/e$.

Proof: Let k denote the number of such polynomials. We use the inclusion-exclusion principle to count the number of elements of $\mathrm{GF}(q^e)$ that do not lie in any maximal

Algorithm 4: `TwoTimesFour`(X, type)

`/*` X `is a generating set for the perfect classical group` G `in odd characteristic, of type SL or Sp or SU, in dimension a multiple of 4. Return standard generating set` Y_0 `for a copy of` $\mathrm{SL}(2,q) \wr S_{d/2} \leq G$`, the SLPs for the elements of` Y_0`, the change-of-basis matrix, and generators for centraliser of involution` k `defined in line 14.` `*/`

1 **begin**
2 $d :=$ the rank of the matrices in X;
3 if $d \leq 4$ then return `BaseCase` (X, *type, false*);
4 $q :=$ the size of the field over which these matrices are defined;
5 if *type* = SU then $q := q^{1/2}$;
6 Find by random search $g \in G := \langle X \rangle$ of even order such that g powers to an involution h with eigenspaces of dimension $n = d/2$;
7 Let n be the dimension of the $+1$-eigenspace of h;
8 Find generators for the projective centraliser C of h in G and identify an element g of C that interchanges the two eigenspaces;
9 In the derived subgroup C' of C find a generating set X_1 for one of the direct factors of C';
10 $(s_1, \delta_1, u_1, v_1) :=$ `TwoEven`(X_1, type);
11 Let $X_2 = X_1^g$;
12 Conjugate all elements of $(s_1, \delta_1, u_1, v_1)$ by g to obtain solution $(s_2, \delta_2, u_2, v_2)$ for X_2;
13 Let $(e_1, f_1, \ldots, e_n, f_n, e_{n+1}, f_{n+1}, \ldots, e_d, f_d)$ be the concatenation of the hyperbolic bases constructed in lines 10 and 12;
14 $k := (\delta_1^{(q-1)/2})^{v_1^{-1}} \delta_2^{(q-1)/2}$;
15 Find generators for the centraliser D of k in G;
16 In the derived subgroup D' of D find a generating set X_3 for the direct factor that acts faithfully on $\langle e_n, e_{n+1}, f_n, f_{n+1} \rangle$;
17 In $\langle X_3 \rangle$ find the permutation matrix $j = (e_n, e_{n+1})(f_n, f_{n+1})$;
18 $v := v_1 j v_2$;
19 return (s_1, δ_1, u_1, v), the change-of-basis matrix, and X_3.
20 **end**

Algorithm 5: `TwoTwiceOdd`$(X, type)$

`/*` X `is a generating set for the perfect classical group` G `in odd characteristic, of type SL or Sp or SU, in twice odd dimension. Return standard generating set` Y_0 `for a copy of` $\mathrm{SL}(2,q) \wr S_{d/2} \leq G$`, the SLPs for the elements of` Y_0`, the change-of-basis matrix, and generators for centraliser of involution` k `defined in line 13.` `*/`

1 **begin**

2 $d :=$ the rank of the matrices in X;

3 if $d \leq 4$ then return `BaseCase` (X, *type, false*);

4 $q :=$ the size of the field over which these matrices are defined;

5 if *type* = SU then $q := q^{1/2}$;

6 Find by random search $g \in G := \langle X \rangle$ of even order such that g powers to an involution h with eigenspaces of dimension 2 and $d-2$.

7 Let n be the dimension of the $+1$-eigenspace of h;

8 Find generators for the centraliser C of h in G;

9 In the derived subgroup C' of C find generating sets X_1 and X_2 for the direct factors of C' where X_2 centralises the eigenspace of dimension 2;

10 $(s_1, \delta_1, u_1, v_1) :=$ `TwoTwiceOdd`$(X_1, type)$;

11 $(s_2, \delta_2, u_2, v_2) :=$ `TwoTimesFour`$(X_2, type)$;

12 Let $(e_1, f_1, \ldots, e_n, f_n, e_{n+1}, f_{n+1}, \ldots, e_d, f_d)$ be the concatenation of the hyperbolic bases constructed in lines and 10;

13 $k := (\delta_1^{(q-1)/2})^{v_1^{-1}} \delta_2^{(q-1)/2}$;

14 Find generators for the centraliser D of k in G;

15 In the derived subgroup D' of D find a generating set X_3 for the direct factor that acts faithfully on $\langle e_n, e_{n+1}, f_n, f_{n+1} \rangle$;

16 In $\langle X_3 \rangle$ find the permutation matrix $j = (e_n, e_{n+1})(f_n, f_{n+1})$;

17 $v := v_1 j v_2$;

18 return (s_1, δ_1, u_1, v), the change-of-basis matrix, and X_3.

19 **end**

Algorithm 6: `TwoEven`$(X, type)$

```
/* X is a generating set for the perfect classical group G in odd
   characteristic, of type SL or Sp or SU, in even dimension.
   Return standard generating set Y_0 for a copy of SL(2,q) ≀ S_{d/2} ≤ G,
   the SLPs for the elements of Y_0, the change-of-basis matrix, and
   generators for centraliser of involution.                          */
1 begin
2     d := the rank of the matrices in X;
3     return TwoTwiceOdd(X, type);
4   return TwoTimesFour(X, type);
end
```

subfield containing GF(q), and divide this number by e, since every irreducible monic polynomial of degree e over GF(q) corresponds to exactly e such elements. Thus

$$k = \frac{q^e - \sum_i q^{e/p_i} + \sum_{i<j} q^{e/p_i p_j} - \cdots}{e}$$

where $p_1 < p_2 < \cdots$ are the distinct prime divisors of e. Thus

$$\begin{aligned} k &= q^e \prod_i (1 - q^{-(1-1/p_i)e})/e \\ &> q^e \prod_{j=1}^{\infty} (1 - q^{-j})/e \\ &> q^e (1 - 1/q)/e. \end{aligned}$$

Clearly $k < (q^e - 1)/e$. The result follows. □

Lemma 6.3 *Let $e > d/2$. The proportion of elements of* GL(d, q) *whose characteristic polynomial has an irreducible factor of degree e is $(1/e)(1 + O(1/q))$. More precisely, there is a universal positive constant c such that the proportion is always between $(1/e)(1 - c/q)$ and $1/e$.*

Proof: Let the characteristic polynomial of $g \in \mathrm{GL}(d, q)$ have an irreducible factor $h(x)$ of degree e. Then $\{w \in V : w.h(g) = 0\}$ spans a subspace of V of dimension e. It follows that the number of elements of GL(d, q) of the required type is $k_1 k_2 k_3 k_4 k_5$ where k_1 is the number of subspaces of V of dimension e, k_2 is the number of irreducible monic polynomials of degree e over GF(q), k_3 is the number of elements of GL(e, q) that have a given irreducible characteristic polynomial, and k_4 is the order of GL$(d - e, q)$, and k_5 is $q^{e(d-e)}$. In more detail,

$$k_1 = \frac{(q^d - 1)(q^d - q) \cdots (q^d - q^{e-1})}{(q^e - 1)(q^e - q) \cdots (q^e - q^{e-1})}$$

$$\begin{aligned} k_3 &= \frac{(q^e-1)(q^e-q)\cdots(q^e-q^{e-1})}{(q^e-1)} \\ k_4 &= (q^{d-e}-1)(q^{d-e}-q)\cdots(q^{d-e}-q^{d-e-1}). \end{aligned}$$

The formula for k_3 arises by taking the index in $\mathrm{GL}(e,q)$ of the centraliser of an irreducible element, this centraliser being cyclic of order q^e-1. The formula of k_2 is given in Lemma 6.2. Hence $k_1k_2k_3k_4k_5 = |\mathrm{GL}(d,q)| \times k_2/(q^e-1)$. The result follows. □

Lemma 6.4 *Let $e \in (d/3, d/2]$. The proportion of elements of $\mathrm{GL}(d,q)$ whose characteristic polynomial has an irreducible factor of degree e is*

$$(e^{-1} - \frac{1}{2}e^{-2})(1+O(1/q)).$$

More precisely there are universal positive constants c_1 and c_2, with $c_1 < c_2$, such that the proportion always lies between $(e^{-1} - \frac{1}{2}e^{-2})(1-c_1/q)$ and $(e^{-1} - \frac{1}{2}e^{-2})(1-c_2/q)$.

PROOF: In the proof of **??** the fact that $e > d/2$ was only used to ensure that the characteristic polynomial of an element $g \in G = \mathrm{GL}(d,q)$ has only one irreducible factor of degree e. In the case of the present lemma, let S_1 denote the number of elements of G whose characteristic polynomial has two distinct irreducible factors of degree e; let S_2 denote the number of elements of G whose characteristic polynomial has a repeated factor of degree e, and let S_3 denote the number of elements of G whose characteristic polynomial has exactly one irreducible factor of degree e. If the proof of **??** is repeated in the present context the result $\frac{1}{2}S_1 + S_2 + S_3 = e^{-1}(1+O(1/q))$ is obtained, as elements contributing to S_1 are counted twice. A similar argument shows that $S_1 = e^{-2}(1+O(1/q))$, and the result follows. □

Lemma 6.5 *The results of Lemmas* 6.3 *and* 6.4 *hold if* $\mathrm{GL}(d,q)$ *is replaced by* $\mathrm{SL}(d,q)$.

PROOF: We first consider the case $e \neq d/2$. We consider matrices that preserve one or (as in Lemma 6.4) two submodules of dimension e. In the case of $\mathrm{SL}(d,q)$ the action on the quotient module (of dimension $d-e$ or $d-2e$) has determinant dictated by the requirement that we are working in $\mathrm{SL}(d,q)$. But there are exactly as many elements of $\mathrm{GL}(d-e,q)$ having one non-zero determinant as another, and similarly for $\mathrm{GL}(d-2e,q)$. Hence the proportion of elements satisfying the condition of either lemma is exactly the same in both $\mathrm{SL}(d,q)$ and $\mathrm{GL}(d,q)$.

If $e = d/2$, so we are in the case of Lemma 6.4, the argument is slightly more complicated. Now the characteristic polynomial of g is the product of two irreducible factors of degree e. If the first is chosen at random, the second must be chosen with a given constant term, because the product of the constant terms is 1, since this is the determinant of g. We can partition the irreducible polynomials of degree e into $q-1$ parts, where the partition is defined by the constant term. We may ignore the

case when the two irreducible factors are equal, and hence suppose that the number of elements in any two $\mathrm{SL}(2e, q)$ conjugacy classes under discussion are equal. The conjugacy classes are determined by the corresponding characteristic polynomials, so it suffices to prove that the number of irreducible polynomials in any two parts differ by a factor of the form $1 + c/q$ for some absolute constant c. In other words, we may work within the multiplicative group $\mathrm{GF}(q^d)^\times$, and consider the number of elements in this group of given norm over $\mathrm{GF}(q)$ that do not lie in any proper subfield that contains $\mathrm{GF}(q)$. But the proportion of elements of $\mathrm{GF}(q^e)$ that lie in a proper subfield that contains $\mathrm{GF}(q)$ is less than c/q for some universal constant c; so we may ignore such elements. Clearly the remaining elements are evenly divided amongst the different values of the norm, these corresponding to the cosets of the group of elements of norm 1.

□

We now obtain a lower bound for the proportion of $g \in \mathrm{SL}(d, q)$ such that g has even order $2n$, and g^n has an eigenspace with dimension in a given range. To perform this calculation, we consider the cyclic groups C_{q^e-1} of order $q^e - 1$. If n is an integer, we write $v_2(n)$ for the 2-adic value of n.

Lemma 6.6 *If $v_2(m) = v_2(n)$ then $v_2(q^m - 1) = v_2(q^n - 1)$.*

PROOF: It suffices to consider the case where $m = kn$, and k is odd. Then $(q^m - 1)/(q^n - 1)$ is the sum of k powers of q, and so is odd. □

Lemma 6.7 *If $u < v$ then $v_2(q^{2^u} - 1) < v_2(q^{2^v} - 1)$, and if $u > 0$ then $v_2(q^{2^u} - 1) = v_2(q^{2^{u+1}} - 1) - 1$.*

PROOF: Observe that $(q^{2^{u+1}} - 1)/(q^{2^u} - 1) = q^{2^u} + 1$ which is even. Now $v_2(q^{2^u} - 1) > 1$ as $u > 0$. It follows that $v_2(q^{2^u} + 1) = 1$. □

Theorem 6.8 *For some absolute constant c, the proportion of $g \in \mathrm{SL}(d, q)$ of even order, such that a power of g is an involution with eigenspaces of dimensions in the range $(d/3, 2d/3]$, is at least c/d.*

PROOF: Let 2^k be the unique power of 2 in the range $(d/3, 2d/3]$. By Lemma 6.5 it suffices to prove that if $g \in \mathrm{SL}(d, q)$ has an irreducible factor of degree 2^k then the probability that g has the required property is bounded away from 0 by some positive constant.

Let $\{W_i : i \in I\}$ be the set of module composition factors of V under the action of $\langle g \rangle$. Let n_i be the order of the image of g in $\mathrm{GL}(V_i)$, and set $w_i = v_2(n_i)$, and $w = \max_i(w_i)$, and $d_i = \dim(W_i)$. If $w > 0$, then g has even order $2n$ say, and in this

case the -1-eigenspace of $h := g^n$ has dimension $\sum d_i$, where the sum is over those values of i for which $w_i = w$.

Suppose now that the characteristic polynomial of g has exactly one irreducible factor of degree 2^k. By renumbering if necessary we may assume that $d_1 = 2^k$. Set $x = v_2(q^{2^k} - 1)$. The probability that $w_1 = x$ is slightly greater than $1/2$. This is because the action of g on W_1 embeds g at random in $\mathrm{GF}(q^{2^k})$, which is a cyclic group of order an odd multiple of 2^x. The distribution of possible values of g is uniform among those elements that do not lie in a proper subfield of $\mathrm{GF}(q^{2^k})$. But non-zero elements of such subfields do not have order a multiple of 2^x. If $w_1 = x$ then necessarily $w_i < w_1$ for all $i > 1$, and $w_1 = w$. It follows that g will then have even order, and that a power of g will be an involution whose -1-eigenspace will have dimension exactly 2^k. It is clear that the elements of $\mathrm{SL}(d, q)$ whose characteristic polynomials have two irreducible factors of degree 2^k may be ignored, and the result follows. □

Corollary 6.9 *Such an element g in $\mathrm{SL}(d, q)$ can be found with at most $O(d(\xi + d^3 \log q))$ field operations, where ξ is the cost of constructing a random element.*

Proof: Theorem 6.8 implies that a search of length $O(d)$ will find such an element g. Lemma 6.1 implies that we can afford to discard elements whose characteristic polynomials have repeated roots.

In $O(d^3)$ field operations the characteristic polynomial $f(t)$ of g can be computed (see [17, Section 7.2]); in $O(d^2 \log q)$ field operations it can be factorised as $f(t) = \prod_{i=1}^{k} f_i(t)$, where the $f_i(t)$ are irreducible (see [30, Theorem 14.14]).

Following the notation of the proof of Theorem 6.8, we may take W_i to be the kernel of $f_i(g)$. It remains to calculate w_i. Let m_i be the odd part of $q^{d_i} - 1$, where d_i is the degree of f_i. Now compute $s := (f_i(t)) + t^{m_i}$ in $\mathrm{GF}(q)[t]/(f_i(t))$, and iterate $s := s^2$ until s is the identity. The number of iterations determines w_i, and it is now easy to determine whether or not g satisfies the required conditions. All of the above steps may be carried out in at most $O(d^3 \log q)$ field operations. □

Our next objective is an algorithm to construct an involution in $\mathrm{SL}(d, q)$ with eigenspaces of equal dimension. This necessarily presupposes that d is a multiple of 4. We use this in Algorithm `TwoEven`.

We describe a recursive procedure to construct an involution in $\mathrm{SL}(d, q)$ whose -1-eigenspace has a specified even dimension e.

1. Search randomly for an element g of even order that powers to an involution h_1 satisfying the conditions of Theorem 6.8.

2. Let r and s denote the ranks of the -1- and $+1$-eigenspaces of h_1.

3. If $r = e$ then h_1 is the desired involution.

4. Consider the case where $s \leq e < r$. Construct the centraliser of h_1, and by powering, obtain generators for the special linear group S_- on the -1-space, and S_- acts as the identity on the $+1$-eigenspace of h_1. By recursion on d, an involution can be found in S_- whose -1-eigenspace has dimension e.

5. Consider the case where $e \leq \min(r, s)$. If $r \leq s$ then construct the centraliser of h, and by powering, obtain generators for the special linear group S_- on this -1-eigenspace, and S_- acts as the identity on the $+1$-eigenspace. By recursion on d, an involution can be found in S_- whose -1-eigenspace has dimension e. Similarly, if $s < r$ then construct S_+, and search in S_+ for an involution whose -1-eigenspace has dimension e.

6. Consider the case where $s \geq e > r$. Construct the centraliser of h, and obtain generators for the special linear group S_+ on the $+1$-eigenspace of h_1, where S_+ acts as the identity on the -1-eigenspace. Now an involution h_2 is found recursively in S_+ whose -1-eigenspace has dimension $e - r$. Then $h_1 h_2$ is an involution of the required type.

7. Finally consider the case where $e \geq \max(r, s)$. This is identical to the last case.

The recursion is founded trivially with the case $d = 4$.

Lemma 6.10 *Using this algorithm, an involution in* $\mathrm{SL}(d, q)$ *can be constructed with* $O(d(\xi + d^3 \log q))$ *field operations that has its* -1*-eigenspace of any even dimension in* $[0, d]$.

PROOF: Corollary 6.9 implies that h_1 can be constructed with at most $O(d(\xi + d^3 \log q))$ field operations. We shall see in Sections 7 and 10 that generators for S_- and S_+ can be constructed with $O(d^4)$ field operations. Thus the above algorithm requires $O(d(\xi + d^3 \log q))$ field operations, plus the number of field operations required in the recursive call. Since the dimension of the matrices in a recursive call is at most $2d/3$, the total complexity is as stated. □

We now consider the other classical groups. If $h(x) \in \mathrm{GF}(q)[x]$ is a monic polynomial with non-zero constant term, let $\tilde{h}(x) \in \mathrm{GF}(q)[x]$ be the monic polynomial whose zeros are the inverses of the zeros of $h(x)$. Hence the multiplicity of a zero of $h(x)$ is the multiplicity of its inverse in $\tilde{h}(x)$ so that $h(x)\tilde{h}(x)$ is a symmetric polynomial. We start with this analogue of Lemma 6.3.

Lemma 6.11 *Let* $m > n/2$. *The proportion of elements of* $\mathrm{Sp}(2n, q)$ *whose characteristic polynomial has a factor* $h(x)$ *where* $h(x)$ *is irreducible of degree* m *and* $h(x) \neq \tilde{h}(x)$ *is* $(1/m)(1 + O(1/q))$. *Moreover there are universal positive constants* $c_1 < c_2$ *such that for all* d *and* q *the proportion lies between* $(1 - c_1/q)m^{-1}$ *and* $(1 - c_2/q)m^{-1}$.

PROOF: Let $g \in \mathrm{Sp}(2n,q)$ act on the natural module V, and let $h(x)$ be an irreducible factor of degree m of the characteristic polynomial $f(x)$ of g. By Lemma 6.1 we may assume that $f(x)$ has no repeated factor. Let V_0 be the kernel of $h(g)$. Since $h(x) \neq \tilde{h}(x)$ it follows that V_0 is totally isotropic. Also $\tilde{h}(x)$ is a factor of $f(x)$, and if V_1 is the kernel of $\tilde{h}(x)$ then V_1 is totally isotropic. Since $h(x)$ and $\tilde{h}(x)$ divide $f(x)$ with multiplicity 1, V_0 and V_1 are uniquely determined, and the form restricted to $V_0 \oplus V_1$ is non-singular. Now let $e_1, \ldots, e_m$ be a basis for V_0. A basis $f_1, \ldots, f_m$ for V_1 is then determined by the conditions $B(e_i, f_j) = 0$ for $i \neq j$, and $B(e_i, f_i) = 1$ for all i, where $B(-,-)$ is the symplectic form that is preserved. The matrix for g restricted to V_0 now determines the matrix of g restricted to V_1, since g preserves the form.

Thus the number of possibilities for g is the product $k_1k_2k_3k_4k_5$, where k_1 is the number of choices for V_0, and k_2 is the number of choices for V_1 given V_0, and k_3 is the number of irreducible monic polynomials $h(x)$ of degree m over $\mathrm{GF}(q)$ such that $h(x) \neq \tilde{h}(x)$, and k_4 is the number of elements of $\mathrm{GL}(m,q)$ with a given irreducible characteristic polynomial, and k_5 is the order of $\mathrm{Sp}(2n-2m,q)$. In more detail

$$\begin{aligned}
k_1 &= \frac{(q^{2n}-1)(q^{2n-1}-q)(q^{2n-2}-q^2)\cdots(q^{2n-m+1}-q^{m-1})}{(q^m-1)(q^m-q)(q^m-q^2)\cdots(q^m-q^{m-1})} \\
k_2 &= q^{(2n-m)+(2n-m-1)+(2n-m-2)+\cdots+(2n-2m+1)} \\
k_3 &\sim q^m/m \\
k_4 &= \frac{(q^m-1)(q^m-q)(q^m-q^2)\cdots(q^m-q^{m-1})}{q^m-1} \\
k_5 &= q^{(n-m)^2}\prod_{i=1}^{n-m}(q^{2i}-1).
\end{aligned}$$

These results are obtained as follows. For k_1, we count the number of sequences of linearly independent elements $(e_1, e_2, \ldots)$ such that each is orthogonal to its predecessors, and divide by the order of $\mathrm{GL}(m,q)$. For k_2, we observe that there is a 1-1 correspondence between the set of candidate subspaces for V_1 and the set of sequences $(f_1, f_2, \ldots, f_n)$ of elements of V such that each f_j satisfies n linearly independent conditions $B(e_i, f_j) = 0$ for $i \neq j$, and $B(e_j, f_j) = 1$.

We observe that k_3 is the number of orbits of the Galois group of $\mathrm{GF}(q^m)$ over $\mathrm{GF}(q)$ acting on those $a \in \mathrm{GF}(q^m)$ that do not lie in a proper subfield containing $\mathrm{GF}(q)$, and have the property that the orbit of a does not contain a^{-1}: namely, the equation $a^{-1} = a^{q^i}$ is not satisfied for any i. This last condition is equivalent to the statement that $h(x) \neq \tilde{h}(x)$. A precise formula for k_3 would be rather complex, so we obtain instead the following estimate. If we ignore this last condition, then Lemma 6.2 estimates k_3. Now it is clear that if $a \in \mathrm{GF}(q^e)$ does satisfy the above equation then the norm of a is 1. In other words, the constant term of $h(x)$ is 1. But this is exactly the problem tackled in the second half of the proof of Lemma 6.5, so we find that, for some absolute constants c_1 and c_2, k_3 lies between $(1-c_1/q)q^m/m$ and $(1-c_2/q)q^m/m$).

Hence the product of the k_i is $k_3|\mathrm{Sp}(2n,q)|/(q^m-1)$ and the result follows. □

Lemma 6.12 *Let $m \in (n/3, n/2]$. The proportion of elements of $\mathrm{Sp}(2n, q)$ whose characteristic polynomial has an irreducible factor $h(x)$ of degree m, where $h(x) \neq \tilde{h}(x)$, is $(e^{-1} - \frac{1}{2}e^{-2})(1 + O(1/q))$. More precisely there are universal positive constants c_1 and c_2, with $c_1 < c_2$, such that the proportion always lies between $(e^{-1} - \frac{1}{2}e^{-2})(1 - c_1/q)$ and $(e^{-1} - \frac{1}{2}e^{-2})(1 - c_2/q)$.*

PROOF: Identical to that of Lemma 6.4. □

We can now prove the analogue of Theorem 6.8.

Theorem 6.13 *For some absolute constant $c > 0$, the proportion of $g \in \mathrm{Sp}(2n, q)$ of even order, such that a power of g is an involution with eigenspaces of dimensions in the range $(2n/3, 4n/3]$, is at least c/n.*

PROOF: Given Lemmas 6.11 and 6.12, the proof is essentially the same as that of Theorem 6.8. □

We now turn to the unitary groups.

Theorem 6.14 *For some absolute constant $c > 0$, the proportion of $g \in \mathrm{SU}(d, q^2)$ of even order, such that a power of g is an involution with eigenspaces of dimensions in the range $(d/3, 2d/3]$, is at least c/d.*

PROOF: The proof is essentially the same as that of Theorem 6.8. □

Theorems 6.8, 6.13 and 6.14 provide an estimate of the complexity of finding an involution of the type required as $O(d(\xi + d^3 \log q))$ field operations. While, from a practical point of view, we regard this as conservative estimate, from a theoretical point of view it is adequate: there are other components of the main algorithms with complexity bounded by $O(d^4 \log q)$ field operations.

7 The Bray algorithm

The centraliser of an involution in a black-box group having an order oracle can be constructed using an algorithm of Bray [4]. Elements of the centraliser are constructed using the following result.

Theorem 7.1 *If u is an involution in a group G, and g is an arbitrary element of G, then $[u, g]$ either has odd order $2k + 1$, in which case $g[u, g]^k$ commutes with u, or has even order $2k$, in which case both $[u, g]^k$ and $[u, g^{-1}]^k$ commute with u.*

That these elements centralise u follows from elementary properties of dihedral groups.

Bray [4] also proves that if g is uniformly distributed among the elements of G for which $[u, g]$ has odd order, then $g[u, g]^k$ is uniformly distributed among the elements

of the centraliser of u. If the order of $g[u,g]^k$ is even, then the elements returned are involutions; but if just one of these is selected, then the elements returned within a given conjugacy class of involutions *are independently and uniformly distributed within that class.*

Let $u \in \mathrm{SL}(d,q)$ and let E_+ and E_- denote the eigenspaces of u. We apply the Bray algorithm in the following contexts.

1. We wish to find a generating set for (a subgroup of) the centraliser of u that contains $\mathrm{SL}(E_+) \times \mathrm{SL}(E_-)$.

2. The eigenspaces, E_+ and E_- have the same dimension. We wish to construct the projective centraliser of u. As we observed in Section 2, the centraliser of u contains an element which interchanges the eigenspaces.

The other contexts are similar, but with $\mathrm{SL}(d,q)$ replaced by the other classical groups.

Parker & Wilson [27] prove that, in a simple classical group of odd characteristic and Lie rank r, the probability of the Bray algorithm returning an odd order element is at least $O(1/r)$. More precisely they prove the following.

Theorem 7.2 *There is an absolute constant c such that if G is a finite simple classical group, with natural module of dimension d over a field of odd order, and u is an involution in G, then $[u,g]$ has odd order for at least a proportion c/d of the elements g of G.*

Our estimate of the efficiency of Bray's algorithm relies critically on their result and the following extension.

Corollary 7.3 *Similar estimates hold for* $\mathrm{SX}(d,q)$.

Proof: We use the notation of Theorem 7.2. Consider first $G = \mathrm{SL}(d,q)$. Suppose that the image of $[u,g]$ in $\mathrm{PSL}(d,q)$ has order n, where n is odd, and $[u,g]^n = \alpha I_d$ for some $\alpha \in \mathrm{GF}(q)$. Then u is conjugate to αu, so either $\alpha = 1$, or $\alpha = -1$ and u has eigenspaces of equal dimension. In the first case, we are in the odd case of Bray's algorithm, whether we work in $\mathrm{SL}(d,q)$ or in $\mathrm{PSL}(d,q)$. In the second case we have found an element that interchanges the two eigenspaces of u. This element is uniformly distributed amongst the elements of the centraliser of u, and hence is equally likely to preserve the eigenspaces as to interchange them. Hence the probability that $[u,g]$ has odd order in $\mathrm{SL}(d,q)$ is half the probability that the image of $[u,g]$ has odd order in $\mathrm{PSL}(d,q)$.

Similar observations apply with the other classical groups under consideration. □

Hence, by a random search of length at most $O(d)$, we construct random elements of the centraliser of the involution. The results of [22] imply that (the derived group of) the centraliser is generated by a bounded number of elements.

It remains to consider a stopping criterion: how can we tell when we have a subset of the centraliser that generates a sufficiently large subgroup? In the first of our

two applications, we apply the Niemeyer-Praeger algorithm [24] to the projection of the centraliser onto each factor to deduce that this is a classical group in its natural representation. This algorithm, when applied to a subgroup of $\mathrm{GL}(k, q)$, has complexity at most $O(k^3)$ group operations. If the factors have the same dimension, there is a small possibility that the given elements generate a group that contains a diagonal embedding of $\mathrm{SL}(d/2, q)$ in $\mathrm{SL}(d/2, q) \times \mathrm{SL}(d/2, q)$ but does not contain the full direct product. This case is easily detected. A similar stopping criterion applies for the second application; we can readily detect when an element of the centraliser interchanges the eigenspaces. Again these remarks apply to the other classical groups.

In its black-box application, the algorithm assumes the existence of an order oracle. We do not require such an oracle for linear groups. If a multiplicative upper-bound B for the order of $g \in G$ is available, then we can learn in polynomial time the *exact* power of 2 (or of any specified prime) which divides $|g|$. By repeated division by 2, we write $B = 2^m b$ where b is odd. Now we compute $h = g^b$, and determine (by powering) its order which divides 2^m. In particular, we can deduce if g has even order. If $g \in \mathrm{GL}(d, q)$, then a multiplicative upper bound of magnitude $O(q^d)$ can be obtained for $|g|$ using the algorithms of [29] and [9] in at most $O(d^3 \log q)$ field operations. This is considered further in Section 10. Further, as discussed in [18], the construction of the centraliser of an involution requires only knowledge of such an upper bound.

We conclude that our applications of the Bray algorithm have complexity $O(d(\xi + d^3 \log q))$ field operations.

8 The base cases

Consider the *base cases*: $\mathrm{SX}(d, q)$ where $d \leq 4$. We construct standard generating sets for these groups using specialised constructive recognition algorithms. We summarise the general algorithm for the base cases and then consider its components in more detail.

Algorithm 7: `BaseCase`$(X, type, Complete)$

```
/* X is a generating set for the perfect classical group G in odd
   characteristic, of type SL or Sp or SU, in dimension at most 4.
   If Complete = false then return standard generating set for a
   copy of SL(2,q) ≀ C_2 ≤ G; otherwise return standard generating set
   for G.  Also return the SLPs for the elements of the set, and
   the change-of-basis matrix.                                    */
1  begin
2      d := the rank of the matrices in X;
3      q := the size of the field over which these matrices are defined;
4      If type = SU then q := q^{1/2};
5      if d = 2 then
6          Apply the SL2 algorithm to construct generating set;
7      else
8          Use centraliser-of-involution algorithm to construct generating set;
9      end
10     return standard generating set, SLPs, and change-of-basis;
```

8.1 The involution-centraliser algorithm

The base case encountered most frequently is $\mathrm{SL}(2, q)$ in its natural representation. An algorithm to construct an element of $\mathrm{SL}(2, q)$ as an SLP in an arbitrary generating set is described in [13]. This algorithm requires $O(\log q)$ field operations, and the use of discrete logarithms in $\mathrm{GF}(q)$.

For $\mathrm{SL}(3, q)$ we use the algorithm of [23] to perform the same task. It assumes the existence of an oracle to recognise constructively $\mathrm{SL}(2, q)$ and its complexity is that of the oracle.

We use the involution-centraliser algorithm of [18] to construct SLPs for elements of $\mathrm{SU}(3, q^2)$ and $\mathrm{SX}(4, q)$. We briefly summarise this algorithm. Assume $G = \langle X \rangle$ is a black-box group with order oracle. We are given $g \in G$ to be expressed as an SLP in X. In this description we say that an element of G is "found" if it is known as an SLP in X. First find by random search $h \in G$ such that gh has even order 2ℓ, and $z := (gh)^\ell$ is a non-central involution. Now find, by random search and powering, an involution $x \in G$ such that xz has even order $2m$, and $y := (xz)^m$ is a non-central involution. Note that x has been found, but, at this stage, neither y nor z has been found. Observe that x, y and z are non-central involutions. We construct their centralisers using the Bray algorithm. We assume that we can solve the explicit membership problem in these centralisers. In particular, we find y as an element of the centraliser of x, and z as an element of the centraliser of y, and gh as an element of the centraliser of z. Having found gh, we have found g.

This is a black-box algorithm requiring an order oracle. However, its performance,

and indeed the question of whether it works at all, depends on the isomorphism type of G.

It is instructive to see why this algorithm fails in the case of $\mathrm{Sp}(4,q)$. This group has only one conjugacy class of non-central involutions, namely the class consisting of elements with both the $+1$ and -1-eigenspaces being mutually orthogonal 2-dimensional spaces. The centraliser of x is isomorphic to $\mathrm{SL}(2,q) \times \mathrm{SL}(2,q)$, and this group contains only two non-central involutions, arising from the unique involution in $\mathrm{SL}(2,q)$. A similar remark applies, of course, to z. Thus, if y is an involution that commutes with x and z, then $z = \pm x$, and the probability of this being the case is too low. Hence x cannot be found efficiently by random search.

In the case of $\mathrm{SU}(4,q^2)$ or $\mathrm{SL}(4,q)$, this problem does not arise. The centraliser of the involution u whose matrix with respect to a hyperbolic basis is

$$\begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

contains many involutions: namely

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

and its conjugates in the centraliser.

To avoid the problem with $\mathrm{Sp}(4,q)$, we work in the projective group $\mathrm{PSp}(4,q)$. This gives rise to two cases. Suppose first that $q \equiv 1 \bmod 4$, so that $\mathrm{GF}(q)$ has a primitive 4-th root ω of 1. Then the projective centraliser of u in $\mathrm{Sp}(4,q)$ contains

$$\begin{pmatrix} \omega & 0 & 0 & 0 \\ 0 & -\omega & 0 & 0 \\ 0 & 0 & \omega & 0 \\ 0 & 0 & 0 & -\omega \end{pmatrix}$$

which has many conjugates in this centraliser. If $q \equiv 3 \bmod 4$, then the projective centraliser of u in $\mathrm{Sp}(4,q)$ contains the projective involution

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

The involution-centraliser algorithm reduces the explicit membership test to three explicit membership tests in involution centralisers; but this is an imperfect recursion,

since the algorithm may not be applicable to these centralisers. We do not rely on the recursion; instead we construct explicitly the desired elements of the centralisers, since these are (direct products of) $\mathrm{SL}(2,q)$.

If G is a linear group, the algorithm does not require an order oracle, exploiting instead the multiplicative bound for the order of an element which can be obtained in polynomial time.

For an analysis of the general algorithm, we refer to [18]; in these specialised cases, we deduce the following result.

Lemma 8.1 *Subject to the availability of a discrete log oracle for* $\mathrm{GF}(q)$*, the standard generators for* $SX(d,q)$ *for* $d \leq 4$ *can be constructed in* $O(\log q)$ *field operations.*

8.2 The glue element

In executing either Algorithm `OneMain` or `TwoMain`, each pair of recursive calls generates an instance of the following problem.

Problem 8.2 *Let* V *be the natural module of* $G = \mathrm{SX}(4,q)$*, and let* (e_1, f_1, e_2, f_2) *be a hyperbolic basis for* V*. Given a generating set for* X*, and the involution* u*, where* u *maps* e_1 *to* $-e_1$ *and* f_1 *to* $-f_1$*, and centralises the other basis elements, construct the involution* j *that permutes the basis elements, interchanging* e_1 *with* e_2*, and* f_1 *with* f_2*.*

Of course, j is the permutation matrix used in each algorithm to "glue" v_1 and v_2 together to form v, the long cycle. (See for example l. 16 of `OneEven`.)

We use the following algorithm to construct this element.

1. Construct the projective centraliser H of u in $\mathrm{SX}(4,q)$, using the Bray algorithm.
2. Since H lies between $\mathrm{SL}(2,q) \wr C_2$ and $\mathrm{GL}(2,q) \wr C_2$, we find $h \in \mathrm{SL}(2,q) \wr C_2$ that interchanges the spaces $\langle e_1, f_1 \rangle$ and $\langle e_2, f_2 \rangle$.
3. Then jh lies in $\mathrm{SL}(2,q) \times \mathrm{SL}(2,q)$. By the powering algorithm described in Section 10, we construct the two direct factors, solve in each direct factor for the projection of jh and so construct jh as an SLP. We can now solve for j.

This algorithm requires $O(\log q)$ field operations.

8.3 The final step

We must also perform the final step of Algorithm `OneMain` or Algorithm `TwoMain`: namely, obtain an additional element.

Consider first the case where d is even. For $\mathrm{SL}(d,q)$, the additional element a allows us to construct the d-cycle from two smaller cycles; in the other cases, we construct the additional element t. This additional element is found in $\mathrm{SX}(4,q)$.

If d is odd and $G = \mathrm{SU}(d,q^2)$, then we must also find the element t in $\mathrm{SU}(3,q^2)$.

In all cases, we employ the involution-centraliser algorithm described in Section 8.1. Lemma 8.1 again applies.

9 Exponentiation

A frequent step in our algorithms is computing the power g^n for some $g \in \mathrm{GL}(d,q)$ and integer n.

Sometimes we raise an element to a high power in order to construct an involution, and we may be able to write down this involution without performing the calculation. However, if, for example, we want to construct elements of one direct factor of a direct product of two groups by exponentiation, then we must explicitly compute the required power.

The value of n may be as large as $O(q^d)$. We could construct g^n with $O(\log(n))$ multiplications using the familiar black-box squaring technique, Instead, we describe an algorithm to perform this task which has complexity $O(d^3)$ field operations.

1. Construct the Frobenius normal form of g and record the change-of-basis matrix.

2. ¿From the Frobenius normal form, we read off the minimal polynomial $h(x)$ of g, and factorise $h(x)$ as a product of irreducible polynomials.

3. This form determines a multiplicative upper bound to the order of g. If $\{f_i(x) : i \in I\}$ is the set of distinct irreducible factors of $h(x)$, and if d_i is the degree of $f_i(x)$, then the order of the semi-simple part of g divides $\prod_i q^{d_i} - 1$, and the order of the idempotent part of g can be read off directly. The product of these two factors gives the required upper bound m.

4. If $n > m$ we replace n by $n \bmod m$. By repeated squaring we calculate $x^n \bmod h(x)$ as a polynomial of degree d.

5. This polynomial is evaluated in g to give g^n.

6. Conjugate g^n by the inverse of the change-of-basis matrix to return to the original basis.

We now consider the complexity of this algorithm.

Lemma 9.1 *Let $g \in \mathrm{GL}(d,q)$ and let $0 \le n < q^d$. Then g^n can be computed using the above algorithm with $O(d^3 + d^2 \log d \log\log d \log q)$ field operations.*

PROOF: The Frobenius normal form of g can be computed with $O(d^3)$ field operations [29] and provides the minimal polynomial. The minimal polynomial can be factored in in $O(d^2 \log q)$ field operations [30, Theorem 14.14]. Calculating $x^n \bmod h(x)$ requires $O(\log(n))$ multiplications in $\mathrm{GF}(q)(x)/(h(x))$, at most $O(d^2 \log d \log\log d \log q)$ field operations [30]. Evaluating the resultant polynomial in g requires $O(d)$ matrix multiplications; but multiplying by g only costs $O(d^2)$ field operations, since g is sparse when in Frobenius normal form. Finally, conjugating g by the inverse of the change-of-basis matrix costs a further $O(d^3)$ field operations. □

One should in theory consider the cost of dividing m by n, even though this does not contribute to the number of field operations. However, for our applications, the exponent n is always less than q^d, so reducing m modulo n is unnecessary.

There is no need to prefer one normal form for g to another, provided that the normal form can be computed in at most $O(d^3)$ field operations, the form is sparse, and the minimum polynomial and multiplicative upper bound for the order of g can be determined readily from the normal form.

This algorithm is similar to that of [9] to determine the order of an element of $\mathrm{GL}(d, q)$.

10 Decomposing direct products

Given a generating set X of $G = \mathrm{SX}(e, q) \times \mathrm{SX}(d - e, q)$ we wish to construct a generating set for one or both of the direct factors.

Let $h \in \mathrm{GL}(d, q)$ have a characteristic polynomial whose irreducible factors have degrees d_i for $1 \leq i \leq k$. The *over-order* of h is the least common multiple of the set of integers $q^{d_i} - 1$ for $1 \leq i \leq k$.

Lemma 10.1 *Let $G = \mathrm{SX}(e, q) \times \mathrm{SX}(d - e, q)$ where $2 \leq e \leq d - 2$. Let S denote the set of elements $(g_1, g_2) \in G$ satisfying one of the following:*

(i) *there exists a prime r dividing the order of g_1, but r does not divide either of $q-1$ or the over-order of g_2;*

Then $\#(S)/|G| > c/d$ for some $c > 0$.

Proof: Consider random $g = (g_1, g_2) \in G$. Suppose that some prime r divides the order of g_1, but does not divide $q - 1$. We must assess the probability that r does not divide the over-order of g_2.

Suppose first that $d - e > 2$. By **??** Lemma 2.3, the proportion of elements of $\mathrm{SL}(d-e, q)$ that have a characteristic polynomial that is either irreducible, or that has an irreducible factor of degree $d - e - 1$ is at least $1/(d - e + 1)$.

The probability that g_1 has order not dividing $q - 1$ is at least $1/2$. (If $e = 2$ this estimate is almost precise, and is very conservative otherwise).

If r is any prime dividing the order of g_1, but not dividing $q - 1$, then either r does not divide $q^{d-e} - 1$, or does not divide the order of $q^{d-e-1} - 1$, as these have greatest common divisor $q - 1$. Thus the proportion of elements in S is at least $1/2(d - e + 1) \geq 1/2(d - 1)$.

Now suppose that $d-e = 2$. If $e > 2$ then the proportion of elements g_1 in $\mathrm{SL}(e, q)$ whose characteristic polynomial is irreducible is at least $1/(e+1)$, and so the proportion of elements in S is at least $1/(e + 1) \geq 1/(d - 1)$.

Finally suppose that $e = d-3 = 2$. Then the probability that g_1 has an irreducible characteristic polynomial and g_2 does not is less than $1/4$.

□

We now consider the cost of constructing a direct factor. We first consider the case where e is small. Let ξ denote the number of field operations required in constructing a random element of G.

Lemma 10.2 *If $1 < e \leq \sqrt{d}$ then a subset of $G = \mathrm{SX}(e,q) \times \mathrm{SX}(d-e,q)$ that generates $\mathrm{SX}(e,q) \times \langle 1 \rangle$ can be constructed from a generating set of G with $O(d\xi + d^{5/2} \log q)$ field operations.*

PROOF: By Lemma 10.1, we require $O(d)$ random elements of G to obtain an element in the set S there defined. The characteristic polynomials of g_1 and g_2 can be computed and factorised in $O(d^3 + d^2 \log q)$ field operations, and this gives their over-orders n_1 and n_2. Lemma 6.1 implies that the probability that the g_2 is inseparable is $O(1/q)$, so we may discard such elements. Then the order of g_2 divides its over-order. Computing $g_1^{n_2}$ can be carried out in $O(d^{3/2} + d \log d \log\log d \log q)$ operations, as proved in Section 9. This gives rise to a non-scalar element h of $H = \mathrm{SX}(e,q) \times \langle 1 \rangle$. Conjugates of h by elements of G generate H. The number of conjugates of h that are required is at most the length of a maximal chain of subgroups of H; since $e \leq \sqrt{d}$ this length is $O(d \log(q))$ REFERENCE?. Since conjugating an element of H by an element of G immediately reduces to forming conjugates in $\mathrm{SX}(e,q)$, the cost of constructing a conjugate is $O(d^{3/2})$ field operations. In $O(d^{3/2})$ field operations the algorithm of [24] determines whether or not a given subset of H generates H. Combining these costs gives the stated bound. □

`There is a gap here, between fixed` k `and` $d/3$

We now consider the bigger values of e.

Niemeyer & Paeger [**?**] and [**?**] prove that, with the exception of a few small values of e, a small random subset of $\mathrm{SL}(e,q)$ will, with high probability, contain a subset of size two that generates $\mathrm{SL}(e,q)$ by virtue of the primes dividing the orders of these elements. That is to say, a pair of elements (h,k) will be found, and prime divisors p_h and p_k of the orders of h and k respectively, such that any pair of elements of $\mathrm{SL}(e,q)$ with one having order a multiple of p_h and the other having order a multiple of p_k will generate $\mathrm{SL}(e,q)$. Thus this property of generating $\mathrm{SL}(e,q)$ is not destroyed by raising h and k to powers that are not multiples of p_h and p_k.

By the above analysis we may exclude small values of d.

Niemeyer & Praeger [24] Section 3.2, prove that, with the exception of a few small values of e, a universally bounded number of random elements of $\mathrm{SX}(e,q)$ are required to find a generating set of size 2, each generator being a multiple of a prime that divides $q^k - 1$ for some $k > e/2$, but which does not divide $q^s - 1$ for any $s < k$. The probability that the primes in question divide the order of a random element of $\mathrm{SX}(d-e,q)$ is small. Thus we obtain the following sharper result for the general case.

Lemma 10.3 *If $e > d/3$, then a subset of $G = \mathrm{SX}(e,q) \times \mathrm{SX}(d-e,q)$ that generates $\mathrm{SX}(e,q) \times \langle 1 \rangle$ can be constructed from a generating set of G in $O(d^3)$ field operations.*

PROOF: Take a random element (g_1, g_2) of G. If the characteristic polynomial of g_2 has no repeated factor, then compute its over-order n_2, and construct $g_1^{n_2}$. This process requires $O(d^3)$ field operations and is repeated a bounded number of times. □

11 Analysis of the algorithms

We now analyse the principal algorithms, and estimate the length of the SLPs that express the canonical generators as words in the given generators. The time analysis is based on counting the number of field operations, and the number of calls to the discrete logarithm oracles. Use of discrete logarithms in a given field requires first the setting up of certain tables, and these tables are consulted for each application. The time spent in the discrete logarithm algorithm, and the space that it requires, are not proportional to the number of applications in a given field.

We first consider the costs associated with tasks not previously discussed.

Babai [2] presented a Monte Carlo algorithm to construct in polynomial time nearly uniformly distributed random elements of a finite group. An alternative is the *product replacement algorithm* of Celler *et al.* [8]. That this is also polynomial time was established by Pak [26]. For a discussion of both algorithms, we refer the reader to [28, pp. 26-30].

Seress [28, Theorem 2.3.9] presents a Monte Carlo polynomial time algorithm to construct a generating set for the derived group of a black-box group. We present an alternative.

Lemma 11.1 *If $SX(d, q) \leq G \leq \mathrm{GL}(d, q)$ then the derived group of G can be constructed in time $O(d^3)$ field operations.*

PROOF: `Needs consideration.` The results of [24] prove that, with the exception of a few small values of e, a universally bounded number of random elements of $\mathrm{SX}(e, q)$ are required to find a generating set of size 2, each generator being a multiple of a prime that divides $q^k - 1$ for some $k > e/2$, but which does not divide $q^s - 1$ for any $s < k$. If $g \in \mathrm{SX}(d, q)$ is a ppd-element, then g^{q-1} is also a ppd-element. Hence we can generate the derived group of $\mathrm{SX}(d, q)$, by raising a bounded number of random elements to their $(q-1)$st power. □

We now complete our analysis of the main algorithms.

Theorem 11.2 *The number of field operations carried out in Algorithm* `OneEven` *is at most $O(d(\xi + d^3 \log q)$.*

PROOF: The construction of a hyperbolic basis for a vector space with a given symplectic or hermitian form, as in line 5, can be carried out in $O(d^3)$ field operations [15, Chapter 2].

The proportion of elements of G with the required property in line 6 is at least k/d for some absolute constant k, as proved in Section 6.

The number of field operations required in lines 8 and 14 is $O(d(\xi + d^3 \log q))$, as proved in Section 7.

The recursive calls in lines 10 and 11 are to cases of dimension at most $2d/3$, and hence they increase only a constant factor the number of field operations.

The number of field operations required in lines 9 and 13 is at most $O(d^3 \log q)$, as proved in Section 10.

The result follows. □

The algorithm is Las Vegas. Thus a more precise statement would be that the probability of kd^4 field operations proving insufficient tends to zero exponentially as a function of k. The field operations counted are the operations of elementary arithmetic.

We record the number of calls to the $\mathrm{SL}(2, q)$ construct recognition algorithm and the associated discrete logarithm oracle.

Theorem 11.3 *Algorithm* `OneEven` *generates at most* $4d$ *calls to the discrete logarithm oracle for* $\mathrm{GF}(q)$.

PROOF: `Needs consideration.` Each call to the constructive recognition oracle for SL2 generates three calls to the discrete logarithm oracle for $\mathrm{GF}(q)$ [13]. Let $f(e) = \alpha \cdot e - 6$ denote the number of calls generated by applying `OneEven` to $\mathrm{SL}(e, q)$, where α is some positive constant. Then $f(d) = f(e) + f(d-e) + 2 \cdot 3 = \alpha d - 6$. There are 9 calls to the discrete log oracle for degree 4. Hence the number of calls is at most $4d$. □
Similar results hold for Algorithm `OneOdd` and so Algorithm `OneMain` also has this complexity.

The results of the analysis of Algorithm `TwoMain` are qualitatively similar; however it generates at most $d - 1$ calls to the constructive recognition algorithm for $\mathrm{SL}(2, q)$.

11.1 Straight Line Programs

We now consider the length of the straight line programs (or SLPs) for the standard generators for $\mathrm{SX}(d, q)$ constructed by our algorithms.

An SLP on a subset X of a group G in its simplest form is a string, each of whose entries is either a pointer to an element of X, or a pointer to a previous entry of the string, or an ordered pair of pointers to not necessarily distinct previous entries. Then every entry of the string defines an element of G. An entry that points to an element of X defines that element. An entry that points to a previous entry defines the inverse of the element defined by that entry. An entry that points to two previous entries defines the product, in that order, of the elements defined by those entries.

An SLP of this simple type can then be thought of as defining an element of G, namely the element defined by the last entry, and this element can be computed by computing in turn the elements for successive entries.

The SLP is used by replacing the elements X of G by the elements Y of some group H, where X and Y are in one-to-one correspondence, and then evaluating the element of H that the SLP then defines.

It is easy to arrange for the algorithms that we have described to construct SLPs for the required group elements; that is to say, for the standard generators of $\mathrm{SX}(d, q)$; on the given generating set X for G.

The reason for using SLPs rather than words in X is that, as successive multiplications are carried out, the length of the corresponding words in X can grow exponentially as a function of the number of multiplications performed, whereas the SLP will only grow linearly.

There are problems with this simple type of SLP.

First we need to replace the second type of node, that defines the inverse of a previously defined element, by a type of node with two fields, one pointing to a previous entry, and one containing a possibly negative integer. The element defined is then the element defined by the entry to which the former field points, raised to the power defined by the latter field. This reflects the fact that we raise group elements to very large powers, and have an efficient algorithm for performing this. Of course it may be convenient to have nodes corresponding to other group-theoretic constructions such as commutators.

Secondly, we should regard an SLP as defining a number of elements of G, and not just one element, so a sequence of nodes may be specified as giving rise to elements of G. Thus we wish to return a single straight line program that defines all the standard generators of $\mathrm{SX}(d, q)$, rather than one straight line program for each of these elements. This avoids duplication when two or more of the standard generators rely on common calculations.

Thirdly, in order to preserve space the structure of an SLP needs to be enhanced to ensure that, when the SLP is evaluated in some other group, the element defined by a node is only calculated when it will be needed later, and is discarded when it it is no longer needed. Discarding the element of H defined by a node when it is no longer required in an evaluation ensures that the space complexity of evaluating an SLP is at worst proportional to the space complexity of the space required to construct the corresponding element (or elements) of G in the first place, given a bound to the space required to store an element of H.

Both algorithms rely on a divide-and-conquer strategy. The first algorithm produces recursive calls to $\mathrm{SX}(e, q)$ and to $\mathrm{SX}(d-e, q)$ where e is approximately $d/2$. The second algorithm reduces to the case when d is a multiple of 4, and then has a single recursive call to $\mathrm{SX}(d/2, q)$. Since the time complexity of the algorithm is greater than $O(d^3)$, for fixed q, the cost in time of the recursive calls is unimportant. This is not the case with the length of the SLPs. The algorithm expends most of its time in random searches; ignoring the construction and testing of random elements that fail to pass the required test, the number of group operations outside recursive calls, including exponentiation as a single operation, becomes constant in the first main algorithm, and $O(\log d)$ in the second, where the involution with eigenspaces of equal dimension that is used when

$d = 4n$ is constructed as a product of $O(\log d)$ involutions.

We now come to the critical point of deciding how the number of trials in a random search for a group element effects the length of an SLP that defines that element. This requires an assumption as to the nature of the random process. We assume that this random process is a stochastic process taking place in a graph whose vertices are defined by a seed, the seed consisting of an array of elements of the group. We refer to this array as 'the seed'. There will be another seed (for a random number generator) that determines which edge adjoining the current vertex in the graph will be followed in the stochastic process.

If no effort is made to improve the situation the length of the SLP will then be increased by a constant amount for every trial, successful or unsuccessful. This constant can, in effect, be regarded as 1 by testing all the group elements constructed in updating the seed. However, this is not good enough for our purposes. We propose the following solution to the problem. When embarking on a search that is expected to require approximately d trials, we record the value of the seed, and repeatedly carry out a random search, using our random process, but returning, after every $c \log d$ steps, for some suitable constant c, to the stored value of the seed, until we succeed. If the vertices in the graph have valency at least v then c should be chosen so that $c \log d$ is significantly less than $\log_v(d)$. With simple versions of product replacement this valency will be at least 50, so in practice the number of steps taken before returning to the stored seed may be taken as a small constant.

Thus in algorithm 1, where the number of recursive calls is $O(d)$, the SLP in question has length approximately $O(d)$. In algorithm 2, where the number of recursive calls is $O(\log d)$, the length of the SLP is approximately $O(\log^2 d)$. However, in each case there are random searches of length $O(d)$ that multiply these estimates in theory, if not in practice, by another factor of $\log d$.

XXXX Small point. In the SLP section we argue for a single SLP to describe all the standard generators. Do we want to use SLP here in the singular or in the plural?

We thus arrive at the following result.

Theorem 11.4 *Counting exponentiation as a single operation, the lengths of the SLP for the standard generators produced by* `OneMain` *is* $O(d \log d)$*; the length produced by* `TwoMain` *is* $O(\log^3 d)$.

12 An implementation

Our implementation of these algorithms is publicly available in Magma. It uses:

- the product replacement algorithm [8] to generate random elements;
- our implementations of Bray's algorithm [4] and the centraliser-of-involution algorithm [18].

The computations reported in Table 1 were carried out using MAGMA V2.12 on a Pentium IV 2.8 GHz processor. The input to the algorithm is $SX(d, q)$. In the column entitled "Time", we list the CPU time in seconds taken to construct the standard generators.

Table 1: Performance of implementation for a sample of groups

Input	Time
$SL_2(8)$	0.2
$SL_2(29)$	0.3
$SL_3(11)$	2.1
$SL_6(2)$	13.1
$Sp(10, 5^{10})$	55.4
$Sp(40, 5^{10})$	2980.4
$SU_8(3^{16})$	22.6
$SU_{20}(5^{12})$	47.6
$SU_{70}(5^2)$	191.3

References

[1] M. Aschbacher, On the maximal subgroups of the finite classical groups, *Invent. Math.*, 76:469–514, 1984.

[2] László Babai, Local expansion of vertex-transitive graphs and random generation in finite groups, *Theory of Computing*, (Los Angeles, 1991), pp. 164–174. Association for Computing Machinery, New York, 1991.

[3] László Babai, William M. Kantor, Péter P. Pálfy and Ákos Seress, Black box recognition of finite simple groups of Lie type by statistics of element orders, *J. Group Theory* **5** (2002), 383–401.

[4] J.N. Bray, An improved method of finding the centralizer of an involution, *Arch. Math. (Basel)* **74** (2000), 241–245.

[5] P.A. Brooksbank, Constructive recognition of classical groups in their natural representation. *J. Symbolic Comput.* **35** (2003), 195–239.

[6] Wieb Bosma, John Cannon, and Catherine Playoust, The MAGMA algebra system I: The user language, *J. Symbolic Comput.*, **24**, 235–265, 1997.

[7] Peter A. Brooksbank and William M. Kantor. On constructive recognition of a black box PSL(d, q). In *Groups and Computation, III (Columbus, OH, 1999)*, volume 8 of *Ohio State Univ. Math. Res. Inst. Publ.*, pages 95–111, Berlin, 2001. de Gruyter.

[8] Frank Celler, Charles R. Leedham-Green, Scott H. Murray, Alice C. Niemeyer and E.A. O'Brien, Generating random elements of a finite group, *Comm. Algebra*, **23** (1995), 4931–4948.

[9] Frank Celler and C.R. Leedham-Green, Calculating the order of an invertible matrix, In *Groups and Computation II*, volume 28 of *Amer. Math. Soc. DIMACS Series*, pages 55–60. (DIMACS, 1995), 1997.

[10] F. Celler and C.R. Leedham-Green. A constructive recognition algorithm for the special linear group. In *The atlas of finite groups: ten years on (Birmingham, 1995)*, volume 249 of *London Math. Soc. Lecture Note Ser.*, pages 11–26, Cambridge, 1998. Cambridge Univ. Press.

[11] Arjeh M. Cohen, Scott H. Murray, and D.E. Taylor. Computing in groups of Lie type. *Math. Comp.* **73**, 1477-1498, 2003.

[12] Marston Conder and Charles R. Leedham-Green. Fast recognition of classical groups over large fields. In *Groups and Computation, III (Columbus, OH, 1999)*, volume 8 of *Ohio State Univ. Math. Res. Inst. Publ.*, pages 113–121, Berlin, 2001. de Gruyter.

[13] M.D.E. Conder, C.R. Leedham-Green, and E.A. O'Brien. Constructive recognition of PSL$(2, q)$. *Trans. Amer. Math. Soc.*, 2006.

[14] Daniel Gorenstein, Richard Lyons, and Ronald Solomon. The classification of the finite simple groups. Number 3. Part I, American Mathematical Society, Providence, RI, 1998.

[15] Larry C. Grove. Classical Groups and Geometric Algebra. AMS Graduate Studies in Math. **39**.

[16] R.M. Guralnick and F. Lübeck. On p-singular elements in Chevalley groups in characteristic p. In *Groups and Computation, III (Columbus, OH, 1999)*, volume 8 of *Ohio State Univ. Math. Res. Inst. Publ.*, pages 113–121, Berlin, 2001. de Gruyter.

[17] Derek F. Holt, Bettina Eick, and Eamonn A. O'Brien. *Handbook of computational group theory.* Chapman and Hall/CRC, London, 2005.

[18] P.E. Holmes, S.A. Linton, E.A. O'Brien, A.J.E. Ryba and R.A. Wilson, Constructive membership testing in black-box groups, preprint.

[19] William M. Kantor and Ákos Seress. Black box classical groups. *Mem. Amer. Math. Soc.*, **149**, 2001.

[20] Charles R. Leedham-Green, The computational matrix group project, in *Groups and Computation*, III (Columbus, OH, 1999), 229–247, Ohio State Univ. Math. Res. Inst. Publ., **8**, de Gruyter, Berlin, 2001.

[21] Martin W. Liebeck and E.A. O'Brien. Finding the characteristic of a group of Lie type. Preprint, 2005.

[22] M.W. Liebeck and A. Shalev. The probability of generating a finite simple group, *Geom. Ded.* **56** (1995), 103–113.

[23] F. Lübeck, K. Magaard, and E.A. O'Brien. Constructive recognition of $\mathrm{SL}_3(q)$. Preprint 2005.

[24] A.C. Niemeyer and C.E. Praeger. A recognition algorithm for classical groups over finite fields, *Proc. London Math. Soc.* **77** (1998), 117–169.

[25] E.A. O'Brien. Towards effective algorithms for linear groups. Preprint, 2005.

[26] Igor Pak. The product replacement algorithm is polynomial. In *41st Annual Symposium on Foundations of Computer Science (Redondo Beach, CA, 2000)*, 476–485, IEEE Comput. Soc. Press, Los Alamitos, CA, 2000.

[27] C.W. Parker and R.A. Wilson. Recognising simplicity in black-box groups. Preprint 2005.

[28] Ákos Seress. *Permutation group algorithms*, volume 152 of *Cambridge Tracts in Mathematics*. Cambridge University Press, Cambridge, 2003.

[29] Arne Storjohann. An $O(n^3)$ algorithm for the Frobenius normal form. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation* (Rostock), 101–104, ACM, New York, 1998.

[30] Joachim von zur Gathen and Jürgen Gerhard, *Modern Computer Algebra*, Cambridge University Press, 2002.

School of Mathematical Sciences
Queen Mary, University of London
London E1 4NS, United Kingdom
United Kingdom
C.R.Leedham-Green@qmul.ac.uk

Department of Mathematics
Private Bag 92019, Auckland
University of Auckland
New Zealand
obrien@math.auckland.ac.nz

Last revised October 2005