

Fast constructive recognition of black box symplectic groups

Peter A. Brooksbank

Abstract

We present an algorithm which, for any given black box group G isomorphic to $\mathrm{Sp}(d, q)$, constructs explicit inverse isomorphisms between G and the standard copy of $\mathrm{Sp}(d, q)$. We also report on the performance of an implementation of this algorithm in the group theory system GAP.

1 Introduction

This is the final paper in a series concerned with constructive recognition of black box classical groups [5, 7] (see also [6, 16]). Of interest in their own right, such algorithms also have important applications to several current computational projects, such as matrix group recognition [18], and the construction of maximal subgroups in finite groups [8]. Applications such as these have increased the demand for high quality computer implementations of constructive recognition algorithms for simple groups. To a greater degree than [5, 7], this paper focuses on practical issues that arise in such implementations.

In a *black box group* elements are encoded (not necessarily uniquely) as binary strings of uniform length. Such groups are equipped with an oracle (the “black box”) to carry out standard group operations. As usual, one assumes that a black box group G is specified by a set of generators. Then, given strings representing elements $g, h \in G$, one can compute (strings representing) gh and g^{-1} , and one can also test whether $g = h$. Black box groups

The author thanks Bill Kantor and Eamonn O’Brien for helpful discussions in the preparation of this paper. He also thanks the referee for suggestions that improved the clarity of the exposition. 2000 *Mathematics Subject Classification*. Primary 20C20, 20C40.

are important because of their great generality (permutation groups and matrix groups are two fundamental examples). Algorithms that compute with black box groups use no information specific to the representation of the input group; they work exclusively with its group structure. Thus black box algorithms are important defaults in many algorithmic settings.

Suppose that a black box group G is known to be isomorphic to a concrete group C . (We assume that C is the *standard copy* of the group; for example, if $G \cong S_8$, then C is the group of all permutations of the set $\{1, \dots, 8\}$.) The goal of a *constructive recognition algorithm* is to construct an *effective isomorphism* $\Psi: C \rightarrow G$. That is, given $c \in C$, there is an efficient procedure to construct $\Psi(c) \in G$, and given $g \in G$, there is an efficient procedure to construct $\Psi^{-1}(g) \in C$.

In practice, constructive recognition algorithms achieve rather more than is suggested above. The data structure that defines an effective isomorphism consists of a set, $\overline{\mathcal{T}}$, of *canonical generators* for $\mathrm{Sp}(d, q)$ together with a corresponding set, \mathcal{T} , of generators for G . To find the image of a given matrix, one writes a *straight-line program (SLP)* (a space-efficient word) from $\overline{\mathcal{T}}$ to the matrix, and then evaluates the SLP from \mathcal{T} . Preimages are obtained in the same manner. Moreover the elements of \mathcal{T} are *constructed* from the original input generators for G using SLPs. Thus, as a by-product of an effective isomorphism, one obtains a procedure to write any given element of G as an SLP from the input generators; this is of crucial importance in applications of such algorithms. (See [14] for another approach to the *constructive membership problem*.)

In [5, 7] constructive recognition algorithms are presented for black box unitary and orthogonal groups. These algorithms are improved versions of an earlier generation of algorithms for black box classical groups [16]. They avoid a recursive call, and hence have improved asymptotic running time over their counterparts in [16]. Furthermore, they admit an oracle for handling subgroups isomorphic to $\mathrm{SL}(2, q)$, and have complexity that is polynomial in the input length modulo this oracle assumption.

In this paper we present an analogous treatment of the symplectic groups. In odd characteristic, symplectic groups are the easiest of the form-preserving classical groups to understand. In characteristic 2, however, certain properties of symplectic groups are best understood using orthogonal groups, via the isomorphism $\mathrm{Sp}(d, 2^k) \cong \Omega(d+1, 2^k)$. This characteristic dichotomy seems to extend to the algorithmic setting: in characteristic 2, the constructive recognition problem for symplectic groups appears to be more challenging, and the resulting algorithms are more complex. We therefore present and analyze the algorithm first for the odd characteristic case. This allows us to focus on practical

aspects of computation with black box groups, and at the same time facilitates a helpful commentary on the more technical algorithms in [5, 7].

Remark 1.1. We assume throughout this paper that the given black box group G is *known* to be a homomorphic image of $\mathrm{Sp}(d, q)$ for known d and q . It is presumed that this knowledge has been obtained by the application of a (faster) *non-constructive* recognition algorithm, such as [2]. Furthermore, at certain stages of our algorithm, we will need to constructively recognize classical subgroups of G of low rank. To avoid costly applications of constructive algorithms with unsuitable inputs, we will always apply non-constructive tests first.

1.1 Statement of results

Let $G = \langle \mathcal{S} \rangle$ be a black box group, and let μ be an upper bound on the time requirement for each group operation in G .

As is standard for algorithms dealing with black box groups, the algorithms presented in this paper are *randomized algorithms*. An algorithm is *Monte Carlo* if the output may be incorrect, but an upper bound on the probability of that can be prescribed by the user. An algorithm is *Las Vegas* if the output is always correct, but there is a possibility that the algorithm will report failure.

A fundamental algorithm due to Babai [1] produces independent, (nearly) uniformly distributed random elements in polynomial time, although a more practical approach is used in computer implementations [10]. Let ξ be an upper bound on the time requirement, per element, for the construction of random elements of G . Since we presume that all of the elements of \mathcal{S} will be involved in the construction of random elements, we assume that $\xi \geq \mu|\mathcal{S}|$.

We hypothesize an oracle to handle computations within subgroups of G isomorphic to $\mathrm{SL}(2, q)$ or $\mathrm{PSL}(2, q)$, for any specified q . Specifically, for a suitable subgroup L , we assume that the oracle returns an effective epimorphism $\Psi_L: \mathrm{SL}(2, q) \rightarrow L$. Let $\chi = \chi(q)$ be an upper bound on the time requirement for each application of this oracle. We assume that $\chi \geq \mu \log q$.

The main theoretical result proved in this paper is the following. As with [5, 7], the complexity of the algorithm compares favourably with its counterpart in [16], even if we take the worst estimate of χ .

Theorem 1.2. *Let $G = \langle \mathcal{S} \rangle$ be a black box group isomorphic to $\mathrm{Sp}(d, q)$ or $\mathrm{PSp}(d, q)$ for known $d \geq 6$ and $q = p^k$, where p is odd. Then there is a Las Vegas algorithm to construct an effective epimorphism $\Psi: \mathrm{Sp}(d, q) \rightarrow G$. The complexity of the algorithm is*

$$O(d^3 \log d \log^4 q + \chi + \xi(d + \log q \log \log q) + \mu d^2 \log^2 q).$$

The Ψ -image of any given matrix in $\mathrm{Sp}(d, q)$ can be constructed in $O(\mu d^2 \log q)$ -time. Given any $g \in G$, one can find $\Psi^{-1}(g) \in \mathrm{Sp}(d, q)$ in $O(\chi d^2 + \mu d^2 \log q)$ -time. The procedures that compute images and preimages are both deterministic.

An analogous algorithm exists when $p = 2$, but the complexity is worse; in fact it is similar to that stated for the orthogonal case in [7, Theorem 1.1]. For rank 2 groups, however, we prove the following result for all fields.

Theorem 1.3. *Let $G = \langle \mathcal{S} \rangle$ be a black box group isomorphic to $\mathrm{Sp}(4, q)$ or $\mathrm{PSp}(4, q)$ for known $q = p^k$. Then there is a Las Vegas algorithm to construct an effective isomorphism $\Psi: \mathrm{Sp}(4, q) \rightarrow G$. The complexity of the algorithm is $O(\chi + \xi \log q)$. The procedures for computing images and preimages under Ψ are exactly the same as those in Theorem 1.2.*

1.2 Overview of the paper

The paper is organized as follows. In Section 2 we summarize the necessary theory of symplectic spaces and their associated groups. The specifics of the hypothesized $\mathrm{SL}(2, q)$ -oracle are detailed in Section 3. Sections 4 and 5 are concerned with constructing a data structure for an effective epimorphism. In Section 4, we consider the rank 2 case, presenting a general purpose algorithm for $\mathrm{Sp}(4, q)$. The general case $d \geq 6$ is considered in Section 5: a detailed algorithm for the odd characteristic case is presented in Sections 5.1 through 5.4; and the obstacles in characteristic 2 are discussed and resolved in Section 5.5. Section 6 is concerned with computing images and preimages under the epimorphism. Finally, in Section 7, we report on our GAP implementation of the algorithm.

2 Preliminaries

In this section we summarise the various properties of symplectic groups that we will need. The reader is referred to [21] for a thorough treatment of the theory of classical groups and their geometric properties. Throughout, V will denote a vector space of dimension d over the finite field $\mathbb{F}_q = GF(q)$, where $q = p^k$ is a prime power, and $\mathrm{GL}(V)$ will denote the group of all invertible linear transformations of V .

2.1 Symplectic spaces and standard bases

Let $(\ , \)$ be a bilinear form on V . A subset $\mathcal{S} \subset \text{GL}(V)$ *preserves* the form if $(v, w) = (v^s, w^s)$ for all $v, w \in V$, $s \in \mathcal{S}$. A space V equipped with a nondegenerate, alternating bilinear form is called a *symplectic space*; the subgroup of $\text{GL}(V)$ consisting of all transformations preserving such a form is a *symplectic group* on V .

Each subspace U of a symplectic space V has an associated subspace $U^\perp = \{v \in V \mid (u, v) = 0 \ \forall u \in U\}$; U is *nonsingular* if $U \cap U^\perp = 0$, and *isotropic* if $U \subseteq U^\perp$. Since $(v, v) = 0$ for all $v \in V$, all *points* (1-spaces) of V are isotropic. Let $m = \max\{\dim(U) : U \text{ is isotropic}\}$ be the *Witt index* of V ; then $\dim(V) = 2m$. A *hyperbolic line* is a nonsingular 2-space, and a *hyperbolic pair* is a basis u, v for such a line satisfying $(u, v) = 1$. A *standard basis* for V is an (ordered) basis $e_1, \dots, e_m, e_{-1}, \dots, e_{-m}$ such that $\langle e_1, \dots, e_m \rangle$ and $\langle e_{-1}, \dots, e_{-m} \rangle$ are (maximal) isotropic spaces, and e_i, e_{-i} is a hyperbolic pair ($1 \leq i \leq m$).

Identify $\text{GL}(V)$ with the group of all invertible $d \times d$ matrices having entries in \mathbb{F}_q . Furthermore suppose that these matrices are written relative to a basis that is standard relative to some fixed nondegenerate, alternating bilinear form $(\ , \)$ associated with V . The (symplectic) group of matrices preserving this form is denoted $\text{Sp}(V)$. We will freely switch between matrices and linear transformations.

A generic symplectic group on a space of dimension d over \mathbb{F}_q (with unspecified invariant form) will be denoted $\text{Sp}(d, q)$.

2.2 The structure of point stabilisers

For each point $x = \langle e \rangle$ of V , there is a subgroup, $T(x)$, of $\text{Sp}(V)$ containing the (symplectic) transvections

$$t(\lambda): v \mapsto v + \lambda(v, e)e, \tag{1}$$

where $\lambda \in \mathbb{F}_q$, inducing the identity on x^\perp and V/x . Hence $T(x) \cong \mathbb{F}_q^+$ has order q . $\text{Sp}(V)$ acts by conjugation on the set of transvection groups as it acts naturally on the set of points of V ; that is, $T(x)^g = T(x^g)$ for all $g \in \text{Sp}(V)$. For distinct points x, y either

- (a) x and y are perpendicular and $T(x)$ and $T(y)$ commute; or
- (b) x and y are not perpendicular and $\langle T(x), T(y) \rangle \cong \text{SL}(2, q)$.

In the latter case $\langle T(x), T(y) \rangle$ is *naturally embedded* in $\text{Sp}(V)$ in the sense that it induces $\text{SL}(2, q)$ on the hyperbolic line $\langle x, y \rangle$ and the identity on $\langle x, y \rangle^\perp$.

The transvection group $T(x)$ is contained in $Q(x) = O_p(\mathrm{Sp}(V)_x)$, the subgroup of $\mathrm{Sp}(V)$ consisting of all transformations of V inducing the identity on x^\perp/x . For any point $y \notin x^\perp$, we have

$$\mathrm{Sp}(V)_x = Q(x) \rtimes \mathrm{Sp}(V)_{x,y},$$

where $(\mathrm{Sp}(V)_{x,y})' \cong \mathrm{Sp}(\langle x, y \rangle^\perp) \cong \mathrm{Sp}(d-2, q)$, and $Q(x)$ acts regularly on the set of all points not contained in x^\perp . It is well known that $\mathrm{Sp}(V)$ is generated by the groups $Q(x)$. In particular, if x and y are not perpendicular, then

$$\langle Q(x), Q(y) \rangle = \mathrm{Sp}(V). \quad (2)$$

Fix a standard basis $e_1, \dots, e_m, e_{-1}, \dots, e_{-m}$ of V . Then $Q(\langle e_1 \rangle)$ consists of all linear transformations of the form

$$r(w, \lambda): e_1 \mapsto e_1, \quad u \mapsto u - (u, w)e_1 \quad \forall u \in \langle e_1, e_{-1} \rangle^\perp, \quad e_{-1} \mapsto e_{-1} + w + \lambda e_1 \quad (3)$$

for $w \in \langle e_1, e_{-1} \rangle^\perp$ and $\lambda \in \mathbb{F}_q$; note $t(\lambda) = r(0, \lambda)$. Products and commutators within $Q(\langle e_1 \rangle)$ behave as follows:

$$r(w_1, \lambda_1) \cdot r(w_2, \lambda_2) = r(w_1 + w_2, \lambda_1 + \lambda_2 - (w_1, w_2)) \quad (4)$$

$$[r(w_2, \lambda_2), r(w_1, \lambda_1)] = r(0, 2(w_1, w_2)) \quad (5)$$

for all $w_1, w_2 \in \langle e_1, e_{-1} \rangle^\perp$ and $\lambda_1, \lambda_2 \in \mathbb{F}_q$. In particular, $Q(\langle e_1 \rangle)$ is elementary abelian when $p = 2$, and $Z(Q(\langle e_1 \rangle)) = [Q(\langle e_1 \rangle), Q(\langle e_1 \rangle)] = T(\langle e_1 \rangle)$ when $p > 2$.

For $\mu \in \mathbb{F}_q^*$, let $s(\mu) \in \mathrm{Sp}(V)_{\langle e_1 \rangle, \langle e_{-1} \rangle}$ send $e_1 \mapsto \mu e_1$, $e_{-1} \mapsto \mu^{-1} e_{-1}$ and induce the identity on $\langle e_1, e_{-1} \rangle^\perp$. Then an action of \mathbb{F}_q is obtained on the $d-2$ -space $Q(\langle e_1 \rangle)/T(\langle e_1 \rangle)$ via $r(w, \lambda)^{s(\mu)} = r(\mu w, \mu^2 \lambda)$. Furthermore, if $\sigma \in (\mathrm{Sp}(V)_{\langle e_1 \rangle, \langle e_{-1} \rangle})'$, then

$$r(w, \lambda)^\sigma = r(w^\sigma, \lambda), \quad (6)$$

and the assignment

$$(r(w, \lambda)T(\langle e_1 \rangle), r(w', \lambda')T(\langle e_1 \rangle)) := (w, w') \quad (7)$$

defines a nondegenerate $(\mathrm{Sp}(V)_{\langle e_1 \rangle, \langle e_{-1} \rangle})'$ -invariant alternating form on $Q(\langle e_1 \rangle)/T(\langle e_1 \rangle)$. Thus $Q(\langle e_1 \rangle)/T(\langle e_1 \rangle)$ is a symplectic module for $(\mathrm{Sp}(V)_{\langle e_1 \rangle, \langle e_{-1} \rangle})'$.

3 The $\text{SL}(2, q)$ -oracle

We assume an oracle to compute effectively with black box groups isomorphic to $\text{SL}(2, q)$. Computer implementations of algorithms that provide the functionality of our hypothesized oracle are discussed in Section 7.1.

It is convenient to think of the oracle rather as three oracles.

The first is an oracle that computes discrete logarithms in \mathbb{F}_q^* : for a fixed generator ρ of \mathbb{F}_q^* , and any given $\lambda \in \mathbb{F}_q^*$, the oracle returns the unique integer $0 \leq n < q - 1$ such that $\rho^n = \lambda$.

The second is an oracle that constructs a suitable data structure for an effective isomorphism:

- $\text{SL2OracleCall}(L, q)$

Given a prime power q and a black box group $L = \langle \mathcal{S} \rangle$, the oracle returns a data structure defining an effective isomorphism $\Psi: \text{SL}(2, q) \rightarrow L$ if $L \cong \text{SL}(2, q)$, and returns **false** if $L \not\cong \text{SL}(2, q)$. The data structure consists of the following components:

1. A field $\mathbb{F}_q = \text{GF}(q)$ and a generator, ρ , of the multiplicative group \mathbb{F}_q^* .
2. A “canonical” generating set $\bar{\mathcal{T}}$ for $\text{SL}(2, \mathbb{F}_q)$ and a new generating set \mathcal{T} for L , together with expressions for the elements of \mathcal{T} as SLPs from \mathcal{S} .
3. A bijection $\bar{\mathcal{T}} \rightarrow \mathcal{T}$ extending to an isomorphism $\Psi: \text{SL}(2, \mathbb{F}_q) \rightarrow L$.

We assume that $\bar{\mathcal{T}}$ consists of the following three matrices:

$$\bar{h} = \begin{bmatrix} \rho & 0 \\ 0 & \rho^{-1} \end{bmatrix}, \quad \bar{t} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad \bar{l} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad (8)$$

The third oracle uses the data structure to write straight-line programs to given elements:

- $\text{SL2SLP}(L, x)$

Given a data structure for $L \cong \text{SL}(2, q)$ obtained from SL2OracleCall , and x , an element of either $\text{SL}(2, q)$ or L , the oracle returns an SLP from $\bar{\mathcal{T}}$ to x if $x \in \text{SL}(2, q)$, or from \mathcal{T} to x if $x \in L$. One may now readily compute images and preimages under Ψ . For example, given $x \in L$, one uses $\text{SL2SLP}(L, x)$ to write an SLP from \mathcal{T} to x , and then evaluates this SLP from $\bar{\mathcal{T}}$ to obtain $\Psi^{-1}(x)$.

4 Algorithms for $\mathrm{Sp}(4, q)$ and $\mathrm{PSp}(4, q)$

In this section we present constructive recognition algorithms for homomorphic images of $\mathrm{Sp}(4, q)$. For these groups, our treatment of the natural representation differs from the black box approach only in the technical details; a more fundamental dichotomy arises from the parity of q .

Various algorithms have already been developed for the 4-dimensional case. For the natural representation, Celler [9] gave the first such algorithm, an improved algorithm is given in [4] by converting to its 5-dimensional orthogonal representation, and an alternative approach is suggested in [11]. For the black box case, an improvement on the algorithm presented in [16] is outlined in [6].

An optimized, general purpose algorithm for $d = 4$ is presented and analyzed in detail here for two reasons. First, it provides a model for the general algorithm that follows. More importantly, the $d = 4$ case is the one that will arise most frequently in practical applications. In particular, we observe that $d = 4$ is the base case of the new recursive approach of Leedham-Green and O'Brien to constructive recognition in the natural representation [17]. At present, their methods work only in odd characteristic, but they will eventually be extended to characteristic 2; the algorithm presented here provides a basis for the recursion in the symplectic case.

The input to the following function is a prime power $q = p^k$, and a group G , assumed to be a homomorphic image of $\mathrm{Sp}(4, q)$ *in any representation*.

- `CRecogniseSp4 (G , q)`

The output is a data structure for an effective epimorphism $\Psi: \mathrm{Sp}(4, q) \rightarrow G$.

The algorithm first constructs the image in G of the subgroup of $\mathrm{Sp}(4, q)$ stabilising a pair of orthogonal hyperbolic lines; this construction is described in Section 4.1. We then obtain generators for the image of a subgroup $Q(x)$ in G (see Section 2.2); this procedure is given in Section 4.2. These constructions yield a new set of generators, \mathcal{T} , for G . Finally, in Section 4.3, a suitable set of preimages, $\overline{\mathcal{T}}$, is obtained. The resulting data structure is compatible with that obtained for the general case in Section 5.

The algorithm is presented first for the case $q > 2$ in Sections 4.1 through 4.4. The case $q = 2$ is discussed separately in Section 4.5.

Element orders: There is no known polynomial-time algorithm to compute the exact order of an element of a black box group. In many practical situations it will be possible

to compute orders exactly, but it is not necessary for our algorithm. We only need to carry out the following two tests:

1. Given $\tau \in G$, find j such that $|\tau| = 2^j m$, where m is odd. If we pre-compute m^* , the odd part of $|\mathrm{Sp}(4, q)|$, then $|\tau^{m^*}| = 2^j$. One can compute τ^{m^*} in time $O(\mu \log q)$ and then determine the value of j by repeated squaring. Note, if τ has even order, then $i(\tau) := \tau^{2^{j-1}m^*}$ is the involution $\tau^{|\tau|/2}$.
2. Given $\tau \in G$ and integer n , test whether τ is a $\mathrm{ppd}^\sharp(p; n)$ -element. In the general case this means that $|\tau|$ is divisible by a *primitive prime divisor* of $p^n - 1$. See [7, Section 2.2.3] for the precise definition of $\mathrm{ppd}^\sharp(p; _)$ -element, and for the relevant tests and facts concerning such elements. In particular, after a pre-processing computation requiring time $O(n^3 \log n \log^4 p)$, one can test whether a given element of G has $\mathrm{ppd}^\sharp(p; n)$ -order in time $O(\mu n \log p)$.

Complexity and reliability: Suppose the probability that a certain event occurs for a single random choice of element of G is $1/N$. Then the probability that this event does not occur at least once within a selection of cN independent random elements of G is at most $(1 - 1/N)^{cN} < e^{-c}$. Hence, with high probability, the event *will occur at least once* within a sample of $O(N)$ random elements. We will use this observation throughout our analysis without further comment.

4.1 Preliminary construction

The first step of the algorithm obtains generators for a certain subgroup K of G .

The stabiliser in $\mathrm{Sp}(V)$ of the orthogonal direct sum of two nonsingular 2-spaces of V is a subgroup isomorphic to $\mathrm{SL}(2, q) \times \mathrm{SL}(2, q)$ (inducing $\mathrm{SL}(2, q)$ on each summand). The following function constructs the image of such a subgroup in the given black box group G .

- $\mathrm{Sp4KeySubgroup}(G, q)$
 1. If q is odd, set $K := \mathrm{Sp4KeyOdd}(G, q)$.
 2. If q is even, set $K := \mathrm{Sp4KeyEven}(G, q)$.
 3. Return K .

As the summary suggests, there is a fundamental divergence in the manner in which K is constructed depending on the parity of q .

The following function assumes that q is odd.

• **Sp4KeyOdd** (G, q)

1. Testing $O(1)$ random elements of G , find τ of even order such that $i := i(\tau) \notin Z(G)$.
2. Compute $K_0 := C_G(i)$ and set $K := K'_0$.
3. Return K .

Lemma 4.1. *The procedure **Sp4KeyOdd** is a Monte Carlo algorithm which, for any odd prime power q and any homomorphic image G of $\mathrm{Sp}(4, q)$, returns the image in G of a subgroup of $\mathrm{Sp}(4, q)$ isomorphic to $\mathrm{SL}(2, q) \times \mathrm{SL}(2, q)$. The complexity of the algorithm is $O(\xi)$.*

Proof. We first address the complexity of the procedure. Step 2 requires the construction of the centralizer in G of an involution. This is done using the method of Bray; as in [14, Theorem 8], K_0 is obtained using $O(1)$ random elements in time $O(\mu \log q)$. Step 2 also requires the construction of derived subgroups. This is done using the algorithm in [20, Theorem 2.3.12]. In the present setting, that algorithm succeeds with high probability using $O(\log q)$ group operations. The stated complexity follows, since $\xi \geq \mu \log q$.

Next we establish the correctness and reliability of the algorithm. One consequence of the work of Parker and Wilson [19] is that, with high probability, a sample of $O(1)$ random elements of G contains an element τ such that $i(\tau)$ is a non-central involution. The derived subgroup of the centralizer of such an involution has the stated structure. \square

Next we consider the case when q is even. The following procedure is based on [16, Section 5.6.1]. Here, however, we avoid unnecessary calls to the $\mathrm{SL}(2, q)$ -oracle in Step 2 by employing a more efficient non-constructive test.

• **Sp4KeyEven** ($G, q = 2^k$)

1. Testing $O(k)$ random elements of G , find a $\mathrm{ppd}^\sharp(p; 2k)$ -element τ .
2. Set $a := \tau^{q-1}$ and repeat the following $O(1)$ times:
 - (a) Choose a random $g \in G$ and set $K_0 := \langle a, a^g \rangle$.
 - (b) Construct $K := K_0^{(\infty)}$, the last term of the derived series of K .
 - (c) Let S a random sample of $8 \lceil \log q \rceil$ random elements of K . If S contains a $\mathrm{ppd}^\sharp(p; 2k)$ -element but not a $\mathrm{ppd}^\sharp(p; 4k)$ -element, then continue to Step 3.
3. Return K .

Lemma 4.2. *The procedure `Sp4KeyEven` is a Monte Carlo algorithm which, for any given $G \cong \mathrm{Sp}(4, 2^k)$, returns the image in G of a subgroup of $\mathrm{Sp}(4, 2^k)$ isomorphic to $\mathrm{SL}(2, 2^k) \times \mathrm{SL}(2, 2^k)$. The complexity of the algorithm is $O(\xi k)$.*

Proof. Noting that $K_0^{(\infty)} = K_0'''$ in Step 2, it is easy to see that `Sp4KeyEven` has the same asymptotic complexity as `Sp4KeyOdd`.

As in [16, Section 5.6.1], the proportion of elements of G having $\mathrm{ppd}^\sharp(p; 2k)$ -order is at least $1/8$, so a suitable τ is found in Step 1 with high probability.

Step 2 constructs a subgroup of G isomorphic to $\mathrm{SL}(2, q) \times \mathrm{SL}(2, q)$, and identifies it using a non-constructive test. For a random element $g \in G$, [16, Lemma 4.12] asserts that $K_0 = \langle a, a^g \rangle$ has the desired structure with probability at least $1/640$. Thus, with high probability, a suitable subgroup K will be examined in Step 2. We claim that the test given in Step 2 accepts suitable subgroups with probability at least $15/16$, and rejects unsuitable ones with the same probability.

An examination of the subgroups of $\mathrm{Sp}(4, q)$ generated by two elements of the same $\mathrm{ppd}^\sharp(p; 2k)$ -order dividing $q + 1$ (see [15, Theorem 5.6]) reveals that the only non-trivial possibilities for $K_0^{(\infty)}$ in Step 2 are as follows:

- (i) $\mathrm{SL}(2, q) \times \mathrm{SL}(2, q)$ (the desired group);
- (ii) $\Omega^-(4, q)$;
- (iii) $q^3 : \mathrm{SL}(2, q)$; and
- (iv) $\mathrm{SL}(2, q)$.

With probability at least $(1 - 1/(2 \log q))^{8 \log q} > 15/16$, a sample of $\lceil 8 \log q \rceil$ random elements from $\mathrm{SL}(2, q) \times \mathrm{SL}(2, q)$ contains at least one of $\mathrm{ppd}^\sharp(p; 2k)$ -order. Hence Step 2 will correctly identify a suitable subgroup with the stated probability. (Note that $\mathrm{SL}(2, q) \times \mathrm{SL}(2, q)$ contains no element of $\mathrm{ppd}^\sharp(p; 4k)$ -order.)

Of the other possible subgroups, only $\Omega^-(4, q)$ contains elements of $\mathrm{ppd}^\sharp(p; 2k)$ -order. However, that group contains elements of $\mathrm{ppd}^\sharp(p; 4k)$ -order in roughly equal proportion to those of $\mathrm{ppd}^\sharp(p; 2k)$ -order in $\mathrm{SL}(2, q) \times \mathrm{SL}(2, q)$; thus such subgroups will be rejected with high probability in Step 2. \square

4.2 Constructing Q

In the previous section we constructed a subgroup K , believed to be a homomorphic image of the stabiliser in $\mathrm{Sp}(4, q)$ of a pair of orthogonal, nonsingular 2-spaces. We next verify this by factoring K into two $\mathrm{SL}(2, q)$ -subgroups, L_1 and L_2 , either as a direct product or as a central product, and then constructing isomorphisms $\Psi_i: \mathrm{SL}(2, q) \rightarrow L_i$ for $i = 1, 2$. A

suitable factoring algorithm is described in detail (and greater generality) in [17, Section 11]. Naturally we invoke `SL2OracleCall` (L_i, q) to construct the isomorphism Ψ_i for $i = 1, 2$. Note that this upgrades the Monte Carlo algorithms in Lemmas 4.1 and 4.2 to Las Vegas algorithms.

Let $\bar{\mathcal{T}}_i = \{\bar{h}_i, \bar{t}_i, \bar{l}_i\}$ and $\mathcal{T}_i = \{h_i, t_i, l_i\}$ denote the new generating sets constructed with these isomorphisms (see Section 3). Set

$$T := \langle t_1, t_1^{h_1}, \dots, t_1^{h_1^{k-1}} \rangle. \quad (9)$$

The following procedure takes as input the given group G , together with the new generating sets for L_1 and L_2 , and constructs an element of $Q = O_p(N_G(T))$ lying outside T .

- **ConstructQ** ($G, \mathcal{T}_1, \mathcal{T}_2$)

1. Find a generator s of G such that $[T, T^s] \neq 1$ and T^s is not contained in L_1 .
2. Set $L := \langle T, T^s \rangle$ and use `SL2OracleCall` (L, q) to construct an effective isomorphism $\text{SL}(2, q) \rightarrow L$.
3. Use `SL2SLP` to construct $h \in L$ of order $q - 1$, normalising T and T^s .
4. Return $u_0 := [h, h_1]$.

Lemma 4.3. *The procedure `ConstructQ` is a deterministic algorithm to construct an element in $Q \setminus T$. The complexity of the algorithm is $O(\chi + \xi)$.*

Proof. The group Q corresponds to some subgroup $Q(x)$ of $\text{Sp}(4, q)$. It follows that Q acts regularly (by conjugation) on the set of all transvection groups that do not commute with T . Since $[T, T^s] \neq 1$, it follows that $Q \rtimes \langle h_1 \rangle = Q \rtimes \langle h \rangle$, and hence that $[h, h_1] \in Q$. The selection of s in Step 1 ensures that $[h, h_1] \notin T$. Since G acts transitively on its transvection groups, any set of generators contains an s behaving as stated. \square

4.3 Computing preimages

In Section 4.2, we constructed $u_0 \in Q \setminus T$. Since L_2 acts naturally as $\text{SL}(2, q)$ on the 2-dimensional module Q/T , we have $Q = \langle T, u_0 \rangle^{L_2}$. As in equation (2), G is generated by Q and Q^{L_1} . Thus, to determine an epimorphism $\Psi: \text{Sp}(4, q) \rightarrow G$, it suffices to obtain suitable preimages for the elements constructed in Sections 4.1 and 4.2.

This is the first place where the algorithm for groups given in their natural representation differs from the generic black box algorithm. We consider the two cases separately.

4.3.1 Natural representation

Here the isomorphism Ψ is effected by applying a suitable change-of-basis.

For $i = 1, 2$ determine the eigenvalues of the element h_i . Since h_i acts as an element of order $q - 1$ on a nonsingular 2-space of the underlying module V , and as the identity on its orthogonal complement, it has eigenvalues $1, \mu_i$ and μ_i^{-1} for some generator μ_i of \mathbb{F}_q^* . Let $e_{\pm i}$ be an eigenvector of h_i corresponding to the eigenvalue $\mu_i^{\pm 1}$. Replacing e_{-i} with $e_{-i}/(e_i, e_{-i})$ we obtain a suitable standard basis of V .

If C is the matrix whose rows are the vectors e_1, e_{-1}, e_2, e_{-2} then the map $X \mapsto C^{-1}XC$ defines our isomorphism $\Psi: \text{Sp}(V) \rightarrow G$.

4.3.2 Black box

If G is a black box group, it is much more complicated to compute suitable preimages of the elements constructed in Sections 4.1 and 4.2.

Let \mathbb{F}_q be the field constructed along with Ψ_2 , and let $V = \mathbb{F}_q^4$. It is convenient here to write matrices relative to a slightly re-ordered standard basis of V that corresponds to the natural actions of L_1 and L_2 . Denote this basis e_1, e_{-1}, e_2, e_{-2} , where L_i acts on the nonsingular 2-space $\langle e_i, e_{-i} \rangle$. We construct $\Psi: \text{Sp}(V) \rightarrow G$ so that it extends $\Psi_2: \text{SL}(\langle e_2, e_{-2} \rangle) \rightarrow L_2$. Thus we may immediately assign preimages to the elements of \mathcal{T}_2 :

$$\bar{h}_2 := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \rho & 0 \\ 0 & 0 & 0 & \rho^{-1} \end{bmatrix}, \quad \bar{t}_2 := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \bar{l}_2 := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix}. \quad (10)$$

(Throughout this paper we will denote the preimage in $\text{Sp}(V)$ of a black box element $g \in G$, under a specified epimorphism, by \bar{g} .)

Preimages of Q : First we construct specific elements of $Q \setminus T$ whose preimages we can write down. Set

$$u := [[u_0, t_2^{l_2}], h_2] \quad \text{and} \quad v := u^{l_2}. \quad (11)$$

Recall the elements $r(w, \lambda)$ defined in equation (3). An easy calculation using equations (4) and (6) reveals that

$$R := [[Q, t_2^{l_2}], h_2] \quad (12)$$

is the image in G of the short root subgroup $R(\langle e_1, e_2 \rangle) = \{r(\alpha e_2, 0) : \alpha \in \mathbb{F}_q\}$ of $Q(\langle e_1 \rangle)$. Since $\langle h_1 \rangle$ centralises L_2 and acts transitively on $R \setminus \{1\}$, we may choose the preimage of u freely from among the elements of the corresponding matrix group $R(\langle e_1, e_2 \rangle)$. We take

$$\bar{u} := \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \text{so that } \bar{v} = \bar{u}^{\bar{t}_2} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}. \quad (13)$$

Preimages of L_1 : First we exchange $t_1 \in T$ for some other $t \in T \setminus \{1\}$ whose preimage we can calculate:

$$t := \begin{cases} [v, u] & \text{if } p > 2 \\ u[t_2, v] & \text{if } p = 2 \end{cases}, \quad \text{so that } \bar{t} = \begin{cases} [\bar{v}, \bar{u}] & \text{if } p > 2 \\ \bar{u}[\bar{t}_2, \bar{v}] & \text{if } p = 2 \end{cases} \quad (14)$$

An easy calculation shows that

$$\bar{t} = \begin{bmatrix} 1 & \lambda & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (15)$$

where $\lambda = 1$ if $p = 2$ and $\lambda = 2$ if $p > 2$. (Remark: This corrects a small oversight in [16, Lemma 5.16] for the case $p = 2$.)

Next we compute the preimage of h_1 under Ψ by comparing the scalar it induces on the short root group R with that induced by h_2 . (Note that h_2 induces ρ on R , and the scalar induced by h_1 on R is therefore a specific power of ρ .)

If $p > 2$ this is done by comparing the transvections $[u^{h_1}, v]$ and $[u^{h_2}, v]$ in T using the isomorphism Ψ_1 . For, if h_1 induces η on R , then $\Psi_1^{-1}([u^{h_1}, v]) = \begin{bmatrix} 1 & \eta \\ 0 & 1 \end{bmatrix}$, whereas $\Psi_1^{-1}([u^{h_2}, v]) = \begin{bmatrix} 1 & \rho \\ 0 & 1 \end{bmatrix}$.

If $p = 2$, then Q is abelian and the commutator approach does not work. Instead we observe that R is conjugate under $\text{Aut}(G)$ to a transvection group. Hence, for any generator s of G such that $[R, R^s] \neq 1$, we have $\langle R, R^s \rangle \cong \text{SL}(2, q)$. For such s , use the $\text{SL}(2, q)$ -oracle to first construct an isomorphism $\text{SL}(2, q) \rightarrow \langle R, R^s \rangle$, and then to compare the scalars induced on R by the two h_i .

If h_1 induces the scalar η on R , then

$$\bar{h}_1 = \begin{bmatrix} \eta^{-1} & 0 & 0 & 0 \\ 0 & \eta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (16)$$

Finally use linear algebra to compute an automorphism α of $\mathrm{SL}(2, q)$ sending $\begin{bmatrix} 1 & 0 \\ \lambda & 1 \end{bmatrix}$ to $\Psi_1^{-1}(t)$ and $\begin{bmatrix} \eta^{-1} & 0 \\ 0 & \eta \end{bmatrix}$ to $\Psi_1^{-1}(h_1)$. Replace Ψ_1 by the isomorphism $\mathrm{SL}(2, q) \rightarrow L_1$ sending $x \mapsto \Psi_1(x^\alpha)$. Then Ψ extends Ψ_1 and Ψ_2 . Replace l_1 with the image under Ψ_1 of the matrix $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$, so that

$$\bar{l}_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (17)$$

Complexity: The total time required to find preimages in the black box case is $O(\chi)$.

4.4 The data structure and total complexity

In the natural representation, we first obtain analogues of the black box elements constructed in Section 4.3.2. Specifically, set $t := t_1$, $u := [[u_0, t_2^{l_2}], h_2]$ and $v := u^{l_2}$. Recall the change-of-basis matrix C defined in Section 4.3.1. For each $y \in \{t, u, v, h_1, l_1, h_2, t_2, l_2\}$, set $\bar{y} := C^{-1}yC$.

The data structure for G consists of the generating sets

$$\bar{\mathcal{T}} := \{\bar{t}, \bar{u}, \bar{v}, \bar{h}_1, \bar{l}_1, \bar{h}_2, \bar{t}_2, \bar{l}_2\} \quad \text{and} \quad \mathcal{T} := \{t, u, v, h_1, l_1, h_2, t_2, l_2\} \quad (18)$$

for $\mathrm{Sp}(4, q)$ and G respectively, together with the data structures for the isomorphisms $\Psi_i: \mathrm{SL}(2, q) \rightarrow L_i$ ($i = 1, 2$). In the natural representation it also includes the change-of-basis matrix C .

Combining the complexity estimates for the procedures in Sections 4.1 through 4.3, we see that the algorithm to construct the data structure for G has total complexity $O(\chi + \xi \log q)$.

Remark 4.4. In order to complete the proof of Theorem 1.3 we must show that the isomorphism defined by this data structure is effective. This is done using the algorithms in Section 6. In that section we will first convert the data structure we have obtained here into a suitable input for the general straight-line program algorithms.

4.5 ‘Brute force’ algorithms in practice (the case $q = 2$)

At various points in [4, 5, 6, 7] it is stated that certain groups of bounded order can be recognised “by brute force”. While such statements are fine for complexity results, one still requires efficient implementations to handle some quite large groups. In practice such exceptions are usually handled as sporadic simple groups and recognised by constructing “standard generators”. These generators may be obtained from the ATLAS of Finite Group Representations (<http://web.mat.bham.ac.uk/atlas/v2.0/>); they differ from the canonical generators that we are using here.

The input to the following function is a group $G = \langle \mathcal{S} \rangle \cong \mathrm{Sp}(4, 2)' \cong A_6$ given in any representation.

- `BruteForceSp42 (G)`

1. Testing up to 20 random elements of G , find $y_0 \in G$ of order 4; set $x := y_0^2$.
2. Testing up to 30 random conjugates of y_0 , find y such that xy has order 5.
3. Apply standard permutation group techniques to construct an isomorphism from $\mathrm{Sp}(4, 2)'$ with generators

$$\bar{x} := \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad \bar{y} := \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

to G , sending $\bar{x} \mapsto x$ and $\bar{y} \mapsto y$.

To see that suitable elements x and y will be found with high probability, note first that one quarter of the elements of A_6 have order 4. If y_0 has order 4 and $x = y_0^2$, then roughly one third of the y_0 -conjugates in A_6 are such that their product with x has order 5.

5 The general algorithm ($d > 4$)

In this section we describe the algorithm for the general case $d \geq 6$. In the interest of clarity we first focus on the (much easier) odd characteristic case. The obstacles for the case $p = 2$ are discussed in Section 5.5, together with indications of how they are overcome.

Our algorithm for black box symplectic groups handles input group G isomorphic to $\mathrm{PSp}(d, q)$ or $\mathrm{Sp}(d, q)$ in exactly the same way. In Section 6 we will construct $Z(G)$ explicitly, and thereby determine the isomorphism type of G . For convenience, we assume in this section that $G \cong \mathrm{Sp}(d, q)$.

The main function takes as input the given black box group G , the dimension d , and the field size $q = p^k$ for $p > 2$.

- `CRecogniseSp` (G, d, q)

The architecture of the algorithm is much the same as that for the 4-dimensional case. The algorithm begins by constructing a certain subgroup, which this time is isomorphic to $\mathrm{Sp}(4, q)$. This preliminary step is described in Section 5.1. Again the next step is to obtain generators for the image, Q , of a subgroup $Q(x)$. This construction is presented in Section 5.2, along with several key algorithms for manipulating the group Q . Finally, in Section 5.3, a new set, \mathcal{T} , of generators for G is constructed, and its preimage, $\overline{\mathcal{T}}$, in $\mathrm{Sp}(V)$ is determined.

5.1 Preliminary construction

In this section we construct a subgroup J of G that acts on the underlying vector space as a *naturally embedded* $\mathrm{Sp}(4, q)$: in its action on the underlying module, J induces $\mathrm{Sp}(4, q)$ on a nonsingular 4-dimensional subspace, and is the identity on its orthogonal complement.

The input to the following function is the given black box group G , the dimension d , and the field size q . The output is a naturally embedded $\mathrm{Sp}(4, q)$ -subgroup J and an element σ of $\mathrm{ppd}^\sharp(p; k(d-2))$ -order.

- `SpKeySubgroup` (G, d, q)
 1. If $q \leq 5$, set $J, \sigma := \mathrm{SpKeySmall}$ (G, q).
 2. If $q > 5$, set $J, \sigma := \mathrm{SpKeyLarge}$ (G, q).
 3. Return J, σ .

The construction follows [16]. However, in the interest of practical utility, fast non-constructive algorithms are employed in both subroutines to test for isomorphism with $\mathrm{Sp}(4, q)$ prior to calling the constructive test `CRecogniseSp4`. The following result can be extracted from [16, Section 5.2.1].

Lemma 5.1. *The procedure `SpKeySubgroup` is a Las Vegas algorithm which, for any given $G \cong \text{Sp}(d, q)$ with $d \geq 6$, constructs a naturally embedded $\text{Sp}(4, q)$ -subgroup J of G . The complexity of the algorithm is*

$$O(d^3 \log d \log^4 q + \chi + \xi(d + \log q \log \log q) + \mu d^2 \log q).$$

We now describe the constructions for each of the two cases. The following function assumes that $q \in \{3, 5\}$ and uses transvections to generate J .

• `SpKeySmall` (G, d, q)

1. Testing $O(d)$ random elements of G , find τ of order $q \cdot \text{ppd}^\#(q; d - 2)$.
2. Set $t := \tau^{q^{(d-2)/2+1}}$.
3. Repeat the following steps $O(1)$ times, until a suitable J is found:
 - (a) Choose g_1, g_2, g_3 at random from G , and set $J := \langle t, t^{g_1}, t^{g_2}, t^{g_3} \rangle$.
 - (b) Use a non-constructive identification algorithm to decide whether J is probably isomorphic to $\text{Sp}(4, q)$ (Remark 1.1). If so, continue to Step 4.
4. Apply `CRecogniseSp4` to J to obtain an isomorphism $\Psi_J: \text{Sp}(V_J) \rightarrow J$, where V_J is a 4-dimensional symplectic space over $\mathbb{F}_q = \text{GF}(q)$.
5. Using a change-of-basis, modify Ψ_J so that $[V_J, \Psi_J^{-1}(t)]$ is spanned by the first standard basis vector of V_J .
6. Return J , the modified Ψ_J , and $\sigma := \tau^q$.

The procedure for arbitrary $q > 5$ is very similar, but uses elements of order dividing $(q - 1)/2$ to generate J . (This is not possible if $q \leq 5$.)

• `SpKeyLarge` (G, d, q)

1. Testing $O(d)$ random elements, find τ of order $\text{ppd}^\#(p; k) \cdot \text{ppd}^\#(p; k(d - 2))$.
2. Set $a := \tau^{q^{(d-2)/2+1}}$.
3. Repeat the following steps $O(1)$ times, until a suitable J is found:
 - (a) Choose a random $g \in G$, and set $J := \langle a, a^g \rangle$.
 - (b) Use a non-constructive identification algorithm to decide whether J is probably isomorphic to $\text{Sp}(4, q)$. If so, continue to Step 4.

4. Apply `CRecogniseSp4` to J to obtain an isomorphism $\Psi_J: \text{Sp}(V_J) \rightarrow J$.
5. As above, modify Ψ_J so that the two 1-dimensional eigenspaces of $\Psi_J^{-1}(a)$ are spanned by the first and third standard basis vectors of V_J .
6. Return J , the modified Ψ_J , and $\sigma := \tau^{q-1}$.

5.1.1 Some useful elements of J

We now use the effective isomorphism Ψ_J obtained above to construct certain key elements that will be used in later constructions.

1. For $0 \leq s < k$, use the algorithm `SpSLP` (presented in Section 6) to construct the transvections

$$t_s := \begin{bmatrix} 1 & \rho^s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Psi_J, \quad (19)$$

and the short root elements

$$r_{1s} := \begin{bmatrix} 1 & 0 & \rho^s & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -\rho^s & 0 & 1 \end{bmatrix} \Psi_J \quad r_{2s} := \begin{bmatrix} 1 & 0 & 0 & \rho^s \\ 0 & 1 & 0 & 0 \\ 0 & -\rho^s & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Psi_J \quad (20)$$

2. Use `SpSLP` again to construct the following elements of J :

$$h := \begin{bmatrix} \rho & 0 & 0 & 0 \\ 0 & \rho^{-1} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Psi_J \quad \text{and} \quad l := \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Psi_J \quad (21)$$

Complexity: By Theorem 1.3 these constructions take $O(\mu d^2 \log^2 q)$ -time.

5.2 Constructing Q

We have just constructed an isomorphism $\Psi_J: \text{Sp}(4, q) \rightarrow J$ for a naturally embedded subgroup J of G . We also constructed an element $\sigma \in G$ that centralises either $t \in J$ or $a \in J$ (see `SpKeySmall` and `SpKeyLarge`, respectively).

Write down the matrix for a nondegenerate, alternating form on $V := \mathbb{F}_q^d$ by taking the usual basis for V to be a standard basis $e_1, \dots, e_m, e_{-1}, \dots, e_{-m}$ relative to the form. Now embed $\text{Sp}(V_J)$ in $\text{Sp}(V)$ in the natural way, by identifying our chosen standard basis of V_J with e_1, e_2, e_{-1}, e_{-2} .

We will eventually construct an isomorphism $\Psi: \text{Sp}(V) \rightarrow G$ that extends Ψ_J .

Let Ψ denote any such isomorphism. Our modification of Ψ_J in Step 4 of `SpKeySmall` and `SpKeyLarge`, and the construction of the elements t_0, \dots, t_{k-1} in Section 5.1.1, ensure that $T := \langle t_0, \dots, t_{k-1} \rangle$ is the Ψ -image of the transvection group $T(\langle e_1 \rangle)$. Furthermore, T is normalised by σ , so that $Q = O_p(N_G(T))$ is also normalised by σ .

In this section we construct generators for Q , and present algorithms to compute effectively both with and within this group.

5.2.1 Effective transitivity of Q

We saw in Section 2 that, for any point x of V , the group $Q(x)$ acts regularly on the set of all points of V that are not perpendicular to x . In this section we present an algorithmic version of this transitivity. This *effective transitivity* is needed in Section 5.2.2, and again in Section 6.

Let φ denote the restriction of Ψ_J to $\text{Sp}(\langle e_1, e_{-1} \rangle)$, so that $\varphi: \text{SL}(2, q) \rightarrow \langle T, T^l \rangle$. Note that if h is the element of order $q-1$ constructed in Section 5.1.1, then $z := h^{(q-1)/2}$, of order 2, generates the center of $\langle T, T^l \rangle$. The input to the following function consists of a transvection group S , not commuting with T , and the isomorphism φ . The output is the unique element of Q conjugating S to T^l .

- `QConjugate` (S, φ)

1. Use the $\text{SL}(2, q)$ -oracle to construct an isomorphism $\text{SL}(2, q) \rightarrow \langle T, S \rangle$.
2. As we did to construct z above, use the isomorphism in Step 1 to construct an element z' of order 2 generating the center of $\langle T, S \rangle$.
3. Set $u := (z'z)^{(p+1)/2}$.
4. Use the isomorphism φ to find the unique $t \in T$ such that $(S^u)^t = T^l$.
5. Return ut .

Lemma 5.2. *`QConjugate` is a deterministic algorithm that constructs the unique element of Q conjugating S to T^l . The complexity of the algorithm is $O(\chi)$.*

Proof. The timing of the procedure is dominated by the uses of the $\mathrm{SL}(2, q)$ -oracle in Steps 1, 2 and 4. It remains to show that $S^u \in \langle T, T^l \rangle$ in Step 3.

Both z and z' centralise T and induce -1 on Q/T . Hence $z'z$ centralises T and Q/T , so that $u = (z'z)^{(p+1)/2} \in Q$.

Since Q acts transitively by conjugation on the set of $\mathrm{SL}(2, q)$ -subgroups of G containing T , there exists $w \in Q$ such that $\langle T, S \rangle^w = \langle T, T^l \rangle$. For any such w we have $(z')^w = z$. Acting as affine transformations of the vector space Q/T , we have $z': v \mapsto -v$ and $w: v \mapsto v + c$ for some vector c . An easy calculation then reveals that $(z'z)^{(p+1)/2} = (z'(z')^w)^{(p+1)/2}$ maps v to $v + c$. Thus u and w induce identical affine transformations of Q/T . It follows that u maps $\langle T, S \rangle$ to $\langle T, T^l \rangle$ as required. \square

5.2.2 Generators for Q

We will now use the element σ found in Section 5.1 to construct generators for Q . First, however, we must modify σ when $q \leq 5$.

Let $\Psi: \mathrm{Sp}(V) \rightarrow G$ denote any isomorphism extending Ψ_J , and let $\bar{\sigma}$ denote $\Psi^{-1}(\sigma)$. For $q \geq 7$, by construction $\bar{\sigma}$ induces the identity on $\langle e_1, e_{-1} \rangle$.

If $q = 3$ or 5 , then $\bar{\sigma}$ induces the identity on some unique hyperbolic line containing $\langle e_1 \rangle$, but not necessarily $\langle e_1, e_{-1} \rangle$. Since $\langle e_{-1} \rangle^{\bar{\sigma}}$ is not perpendicular to $\langle e_1 \rangle$, there is a unique $\bar{u} \in Q(\langle e_1 \rangle)$ such that $\langle e_{-1} \rangle^{\bar{\sigma}\bar{u}} = \langle e_{-1} \rangle$. Hence $\bar{\sigma}\bar{u}$ stabilises $\langle e_1, e_{-1} \rangle$ and $\langle e_1, e_{-1} \rangle^\perp$. Furthermore, since \bar{u} is the identity on $\langle e_1 \rangle^\perp / \langle e_1 \rangle$, it follows that $\bar{\sigma}\bar{u}$ has $\mathrm{ppd}^\sharp(p; k(d-2))$ -order. Hence $(\bar{\sigma}\bar{u})^{q-1}$ is a $\mathrm{ppd}^\sharp(p; k(d-2))$ -element inducing the identity on $\langle e_1, e_{-1} \rangle$.

Algorithmically it is easy to find $u \in Q$ such that $\bar{u} = \Psi^{-1}(u)$ behaves as above. Namely, set $u := \mathbf{QConjugate}(T^{l\sigma}, \varphi)$, where $\varphi: \mathrm{SL}(2, q) \rightarrow \langle T, T^l \rangle$ is the restriction of Ψ_J to $\langle e_1, e_{-1} \rangle$. Now replace σ by $(\sigma u)^{q-1}$.

In all cases we now have an element σ of $\mathrm{ppd}^\sharp(p; k(d-2))$ -order satisfying the following condition: *For any isomorphism $\Psi: \mathrm{Sp}(V) \rightarrow G$ extending Ψ_J , $\Psi^{-1}(\sigma)$ is the identity on $\langle e_1, e_{-1} \rangle$.*

We can now construct our generating set for Q . Set

$$\mathcal{S}_Q := \{ t_s, r_{1s}, r_{2s}^{\sigma^i} : i \in \{0, \dots, d-3\}, s \in \{1, \dots, k\} \} \quad (22)$$

Lemma 5.3. $\langle \mathcal{S}_Q \rangle = Q$.

Proof. It is clear from the construction of \mathcal{S}_Q that $\langle \mathcal{S}_Q \rangle / T$ is a subspace of the \mathbb{F}_q -space Q/T . Since σ acts irreducibly on this space, the result follows. \square

Complexity: The elements of \mathcal{S}_Q are constructed in time $O(\mu d \log q)$.

5.2.3 A form on Q/T

We saw in equation (7) that

$$(r(w_1, \lambda_1)T(\langle e_1 \rangle), r(w_2, \lambda_2)T(\langle e_1 \rangle)) := (w_1, w_2)$$

defines an $(\mathrm{Sp}(V)_{\langle e_1 \rangle, \langle e_{-1} \rangle})'$ -invariant alternating form on $Q(\langle e_1 \rangle)/T(\langle e_1 \rangle)$. We now construct a corresponding invariant bilinear form on Q/T .

- **InvariantForm** (\mathcal{S}_Q, Ψ_J)

1. For $i \in \{1, \dots, d-2\}$ and $j \in \{i+1, \dots, d-2\}$ proceed as follows:

(a) Compute $t_{ij} := [r_{j1}, r_{i1}] \in T$, and

$$\Psi_J^{-1}(t_{ij}) = \begin{bmatrix} 1 & \lambda & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (23)$$

(b) Set $\alpha_{ij} := \lambda/2$ and $\alpha_{ji} := -\lambda/2$.

2. Return the matrix $A = [[\alpha_{ij}]]$.

Lemma 5.4. *Let $L = (N_G(T) \cap N_G(T^l))'$. Then the following hold:*

1. *L is the image in G of $(\mathrm{Sp}(V)_{\langle e_1 \rangle, \langle e_{-1} \rangle})'$ under any isomorphism $\Psi: \mathrm{Sp}(V) \rightarrow G$ extending Ψ_J ; in particular, $L \cong \mathrm{Sp}(d-2, q)$.*

2. ***InvariantForm** is an $O(d^2(\mu+\chi))$ -time algorithm that returns a matrix representing an L -invariant, alternating bilinear form on Q/T .*

Proof. Let $\Psi: \mathrm{Sp}(V) \rightarrow G$ be any isomorphism extending Ψ_J . It is easy to see that $N_G(T)$ (respectively $N_G(T^l)$) is the image under Ψ of $\mathrm{Sp}(V)_{\langle e_1 \rangle}$ (respectively $\mathrm{Sp}(V)_{\langle e_{-1} \rangle}$). The structure of L is now clear.

Fix $i \in \{1, \dots, d-2\}$, and let $r(u_i, \lambda_i)$ denote the preimage of r_{i1} under Ψ . Then, by equation (5), $r(0, 2(u_i, u_j))$ is the preimage of $t_{ij} = [r_{j1}, r_{i1}]$. The result now follows from equation (7). \square

5.3 Computing preimages

Recall that we intend to construct an isomorphism $\Psi: \text{Sp}(V) \rightarrow G$ extending Ψ_J . Since $G = \langle Q, Q^l \rangle$, it suffices to write down suitable preimages in $\text{Sp}(V)$ for our constructed elements of Q . This is achieved using the matrix representing the form on Q/T as follows.

Set $w_1 := e_2, w_2 := e_{-2}$, and use linear algebra to find vectors w_3, \dots, w_{d-2} in the $(d-4)$ -dimensional subspace $\langle e_1, e_2, e_{-1}, e_{-2} \rangle^\perp$ of V such that $(w_i, w_j) = \alpha_{ij}$ for $1 \leq i < j \leq d-2$.

Lemma 5.5. *There is a unique isomorphism $\text{Sp}(V) \rightarrow G$ extending Ψ_J and sending $r(\rho^s w_i, 0) \mapsto r_{is}$ ($1 \leq i \leq d-2, 0 \leq s < k$).*

Proof. Let $\Psi: \text{Sp}(V) \rightarrow G$ be any isomorphism extending Ψ_J , and let $\bar{\sigma}$ denote the element $\Psi^{-1}(\sigma) \in \text{Sp}(V)$. Recall from Section 5.2.2 that $\bar{\sigma} \in \text{Sp}(V)_{e_1, e_{-1}}$.

Any automorphism of $\text{Sp}(V)$ that centralises $\text{Sp}(\langle e_1, e_2, e_{-1}, e_{-2} \rangle)$ is induced by conjugation under an element of $\text{Sp}(V)$ that acts as the identity on $\langle e_1, e_2, e_{-1}, e_{-2} \rangle$. In turn, each such conjugating matrix is uniquely determined by an isometry of $\langle e_1, e_2, e_{-1}, e_{-2} \rangle^\perp$.

Fix $i \in \{1, \dots, d-2\}$ and $s \in \{0, \dots, k-1\}$. As in equation (6), the preimage of r_{is} under Ψ is $r(\rho^s u_i, 0)$ for some $u_i \in \langle e_1, e_{-1} \rangle^\perp$. Furthermore, it follows from the proof of Lemma 5.4 that $(u_i, u_j) = \alpha_{ij}$ for all $1 \leq i, j \leq d-2$. Hence the map $u_i \mapsto w_i$ extends to an isometry of $\langle e_1, e_2, e_{-1}, e_{-2} \rangle^\perp$. Let C be the corresponding change-of-basis matrix. Then the isomorphism $\text{Sp}(V) \rightarrow G$ sending $\bar{g} \mapsto \Psi(C\bar{g}C^{-1})$ is the unique one having the stated property. \square

It follows from the previous lemma that our target isomorphism $\Psi: \text{Sp}(V) \rightarrow G$ is determined by the obvious bijection $\bar{\mathcal{S}}_Q \cup \{\bar{l}\} \rightarrow \mathcal{S}_Q \cup \{l\}$, where

$$\bar{\mathcal{S}}_Q := \{ r(0, r^s), r(\rho^s w_i, 0) : i \in \{1, \dots, d-2\}, s \in \{0, \dots, k-1\} \} \quad (24)$$

Complexity: The matrices in $\bar{\mathcal{S}}_Q$ are determined and written down in time $O(d^3 \log^2 q)$.

5.4 The data structure and total complexity

We now perform all computations within $\text{Sp}(V)$ and duplicate them in G via the bijection $\bar{\mathcal{S}}_Q \cup \{\bar{l}\} \rightarrow \mathcal{S}_Q \cup \{l\}$. The details of these constructions are given in [4, Sections 4.5, 4.6]; we just summarize them here. (Note that the cases $p > 2$ and $p = 2$ are handled identically from this point.)

First use linear algebra to construct a “standard” generating set

$$\bar{\Delta} := \{ r(0, \rho^s), r(\rho^s e_j, 0) : j \in \{\pm 2, \dots, \pm(d-2)/2\}, s \in \{0, \dots, k-1\} \} \quad (25)$$

for $Q(x)$, whose elements have all been obtained using SLPs from $\overline{\mathcal{S}}_Q$. Those SLPs are then evaluated in Q from \mathcal{S}_Q to give a “standard” generating set, Δ , for Q .

Next, commutator relations are used to exchange the generating set $\overline{\Delta} \cup \{\bar{l}\}$ of $\text{Sp}(V)$ for a new set of generators, $\overline{\mathcal{T}}$, of size $O(kd^2)$. Once again the elements of $\overline{\mathcal{T}}$ are obtained using SLPs from $\overline{\Delta} \cup \{\bar{l}\}$, which are then evaluated from $\Delta \cup \{l\}$ to obtain the image, \mathcal{T} , of $\overline{\mathcal{T}}$ in G .

The data structure for the effective isomorphism $\Psi: \text{Sp}(V) \rightarrow G$ consists of the bijection $\overline{\mathcal{T}} \rightarrow \mathcal{T}$, and the data structure constructed along with the isomorphism $\text{Sp}(4, q) \rightarrow J$.

Complexity: The time required to construct the sets $\overline{\mathcal{T}}$ and \mathcal{T} from the sets $\overline{\mathcal{S}}_Q \cup \{\bar{l}\}$ and $\mathcal{S}_Q \cup \{l\}$ is $O(\mu d^2 \log q)$.

Combining all of the timings in Sections 5.1 through 5.4, we get the complexity estimate stated in Theorem 1.2.

5.5 Obstacles in characteristic 2

We have already seen in Section 4 that symplectic groups exhibit fundamentally different behaviour when the characteristic of the defining field is 2. The main reason is that, in characteristic 2, symplectic groups are often more conveniently handled as orthogonal groups via the isomorphism $\text{Sp}(2m, 2^k) \cong \Omega(2m + 1, 2^k)$. This phenomenon presents nontrivial algorithmic difficulties.

In this section we summarise the main obstacles in characteristic 2 and explain how to overcome them. The single case $G \cong \text{Sp}(6, 2)$ is treated as a sporadic simple group, and handled using standard generators, as in Section 4.5. Hence we may assume that $d \geq 8$ if $q = 2$.

5.5.1 Preliminary construction

When $q = 2^k$, the problem of generating a naturally embedded $\text{Sp}(4, q)$ -subgroup of $\text{Sp}(d, q)$ seems to be more difficult than for the odd characteristic case. However, it is relatively easy to generate a naturally embedded low-dimensional subgroup of the isomorphic orthogonal group $\Omega(d + 1, q)$. Additional constructions are then carried out to generate the desired $\text{Sp}(4, q)$ -subgroup.

Let G be the given black box group, isomorphic to $\text{Sp}(d, 2^k)$. The following construction is taken from [16, 5.2.2].

If $k > 1$, we search for an element τ of $\text{ppd}^\sharp(2; k) \cdot \text{ppd}^\sharp(2; k(d-2))$ -order, exactly as in `SpKeyLarge`. Set $a := \tau^{2^{k(d-2)/2+1}}$ and $\sigma := \tau^{2^k-1}$. Then, for a random pair $(g_1, g_2) \in G \times G$, we have $K := \langle a, a^{g_1}, a^{g_2} \rangle \cong \Omega^+(6, 2^k)$ with high probability.

If $k = 1$ (so that $d \geq 8$), we search instead for an element τ of $5 \cdot \text{ppd}^\sharp(2; d-4)$ -order. Set $a := \tau^{2^{(d-4)/2+1}}$ and $\sigma := \tau^5$. Then, for a random $g \in G$, we have $K := \langle a, a^g \rangle \cong \Omega^-(8, 2)$ with high probability.

In both cases we use a fast nonconstructive test to discard choices K that are likely to be unsuitable (see `SpKeySmall`, `SpKeyLarge` and Remark 1.1). For a suitable choice K we then apply an appropriate algorithm to obtain an isomorphism $\Psi_K: \Omega^+(6, 2^k) \rightarrow K$ or $\Psi_K: \Omega^-(8, 2) \rightarrow K$. (Note that $\Omega^+(6, 2^k)$ may be recognised first as $\text{SL}(4, 2^k)$ using the algorithm in [6], and then converted to $\Omega^+(6, 2^k)$; instances of $\Omega^-(8, 2)$ are handled by brute force, as in Section 4.5.)

Unlike the subgroup J constructed in Section 5.1, K plays a fleeting rôle in our algorithm. The main reason for this, beyond the desire for algorithmic uniformity, is that *neither $\Omega^+(6, 2^k)$ nor $\Omega^-(8, 2)$ contain transvections*. In [16, Section 5.3.1], an algorithm is presented that uses K and σ to construct a group of transvections, T , in G , normalized by σ . Once T has been constructed, K is discarded, and we follow the approach taken in `SpKeySmall` for $q = 3$ and 5, constructing $J \cong \text{Sp}(4, q)$ using G -conjugates of T .

5.5.2 Algorithms for Q

Characteristic 2 replacements are needed for the algorithms presented in Section 5.2 to compute with $Q = O_p(N_G(T))$.

Effective transitivity of Q : We need a characteristic 2 version of the function `QConjugate` presented in Section 5.2.1. Once again, we require an element of Q conjugating a given transvection group S , not commuting with T , to T^l .

The $\text{SL}(2, 2^k)$ -subgroups $\langle T, T^l \rangle$ and $\langle T, S \rangle$ used in `QConjugate` are now simple, so the elements z, z' no longer exist. Instead we use the original h of order $q-1$ normalising T and T^l together with a corresponding element h' normalising T and S . Using the $\text{SL}(2, q)$ -oracle, arrange for h and h' to induce the same automorphism of T . Suppose that this common automorphism corresponds to a generator ρ of $\text{GF}(2^k)^*$. Use discrete logarithms in $\text{GF}(2^k)^*$ to compute the integer j such that $\rho^j(\rho-1) = 1$. Then $u := (h^{-1}h')^{h^j}$ conjugates S within $\langle T, T^l \rangle$.

Generators for Q : No modifications are required here. Generators for Q are obtained

as in the odd characteristic case, using σ -conjugates of elements of $O_p(N_J(T))$. Note, however, that σ does not act irreducibly on Q/T when $q = 2$, since it preserves a proper subspace of dimension $d - 4$. In that single case, we only generate Q with high probability (cf. [16, Lemma 5.11]); any such failure will be detected at the next stage of the algorithm when we construct a form on Q/T .

A form on Q/T : The last major obstacle is to devise an alternative method to construct an invariant form on Q/T , and thereby obtain an analogue of the function `InvariantForm` presented in Section 5.2.3. Since Q is abelian in characteristic 2, all of the commutators $[r_{j1}, r_{i1}]$ used to compute the matrix entries α_{ij} in that function are equal to the identity.

We proceed in a manner similar to that described in [7, Procedure 3.21] to compute the scalars α_{ij} . Here, however, things are slightly more straight-forward. As we have now constructed the transvection group T , we can work within the group $K_{ij} := \langle T, T^l, r_{i1}, r_{j1} \rangle$. There are two possibilities for the structure of K_{ij} , corresponding to the orthogonality, or otherwise, of $r_{i1}T$ and $r_{j1}T$ in Q/T . If these vectors are not perpendicular, then the 4-dimensional support of K_{ij} (in its action on the underlying symplectic module V) is nonsingular, and $K_{ij} \cong \text{Sp}(4, 2^k)$. If, on the other hand, they are perpendicular, then the 4-space underlying K_{ij} has a 2-dimensional radical, and K_{ij} is isomorphic to a 2-group extended by $\text{SL}(2, 2^k)$.

We therefore proceed as follows. Choose $O(\log d)$ elements of K_{ij} . If any one of those elements has $\text{ppd}^\sharp(2; 4k)$ -order, then we know $K_{ij} \cong \text{Sp}(4, 2^k)$ and we call `CRecogniseSp4` ($K_{ij}, 2^k$) to construct an isomorphism $\Psi_{K_{ij}}: \text{Sp}(4, 2^k) \rightarrow K_{ij}$ that agrees with Ψ_J on $\langle T, T^l \rangle$. The scalar α_{ij} is then computed within K_{ij} using the isomorphism $\Psi_{K_{ij}}$. If none of the elements have $\text{ppd}^\sharp(2; 4k)$ -order, we conclude that $K_{ij} \not\cong \text{Sp}(4, 2^k)$ and set $\alpha_{ij} := 0$. The correctness and reliability of this procedure is analyzed as in [7, Procedure 3.21].

5.5.3 Completing the data structure.

Sections 5.3 and 5.4 apply to the characteristic 2 case without modification.

6 Straight-line programs

In this section we use the data structures described in Sections 4.4 and 5.4 to define $\Psi: \text{Sp}(V) \rightarrow G$ *constructively*, by presenting algorithms to solve the following problems:

- (a) Given any $\bar{g} \in \text{Sp}(V)$, find $\Psi(\bar{g}) \in G$.

(b) Given any $g \in G$, find $\Psi^{-1}(g) \in \text{Sp}(V)$.

Recall that in each case ($d = 4$ and $d > 4$) the data structure contains a bijection $\overline{\mathcal{T}} \rightarrow \mathcal{T}$ between generating sets of $\text{Sp}(V)$ and G . Problems (a) and (b) are solved by devising algorithms for the following:

(A) Given any $\bar{g} \in \text{Sp}(V)$, write an SLP from $\overline{\mathcal{T}}$ to \bar{g} .

(B) Given any $g \in G$, write an SLP from \mathcal{T} to g .

An algorithm to solve Problem (A) is given in [4, Section 5]. The complexity of this algorithm $O(d^3 \log q + \log^2 q)$.

Constructing $Z(G)$ when q is odd: As an immediate application of (A), we can now determine whether G is simple in the odd characteristic case by constructing $Z(G)$. Write an SLP from $\overline{\mathcal{T}}$ to $-I_d \in \text{Sp}(V)$, and evaluate this SLP from \mathcal{T} to obtain z generating $Z(G)$. (For convenience, if $z \neq 1$, append $-I_d$ to $\overline{\mathcal{T}}$ and z to \mathcal{T} , and extend the bijection to this pair of elements.)

We now turn to Problem (B).

When $d > 4$, the data structure contains an effective isomorphism $\Psi_J: \text{Sp}(4, q) \rightarrow J$. Let $\varphi: \text{SL}(2, q) \rightarrow \langle T, T^l \rangle$ be the restriction of Ψ_J to $\text{Sp}(\langle e_1, e_{-1} \rangle)$. Also $\overline{\mathcal{T}}$ contains the generating set $\overline{\Delta}$ for $Q(x)$, defined in equation (25), and \mathcal{T} contains its image, Δ , in Q .

When $d = 4$, the data structure already contains a suitable isomorphism φ , namely $\Psi_1: \text{SL}(2, q) \rightarrow L_1$. In this case, a suitable analogue of Δ is constructed from \mathcal{T} as follows: For $1 \leq s \leq k$, set $u_k := u^{h_2^{s-1}}$, $v_k := v^{h_2^{s-1}}$ and $t_s = [v_1, u_s]$; then set

$$\Delta := \{t_s, u_s, v_s : 1 \leq s \leq k\}.$$

The pre-image, $\overline{\Delta}$, of Δ is constructed in a similar manner from $\overline{\mathcal{T}}$. Extend $\overline{\mathcal{T}}$ and \mathcal{T} to contain $\overline{\Delta}$ and Δ , respectively.

The following function, which takes as input the generating sets $\overline{\mathcal{T}}$ and \mathcal{T} , the effective isomorphism φ , and any element $g \in G$, solves Problem (B).

• SpSLP ($\overline{\mathcal{T}} \rightarrow \mathcal{T}$, φ , g)

1. Initialize $c := g$.
2. Find $b \in \{1, l\} \cup \Delta^l$ with $[T, T^{cb}] \neq 1$. Set $c := cb$. /* T^c is opposite T */
3. Set $u := \text{QConjugate}(\varphi, T^c)$. Set $c := cul^{-1}$. /* c normalizes T */

4. Set $v := \mathbf{QConjugate}(\varphi, T^{lc})$. Set $c := cv$. /* c normalizes T and T^l */
5. Use φ and discrete logarithms to find j such that c and h^j induce the same scalar on T . Set $c := ch^{-j}$. /* c induces the identity on T */
6. Compute the matrix $\Gamma \in \mathrm{Sp}(d-2, q)$ induced by c on Q/T .
7. Embed Γ in a $d \times d$ matrix $\bar{c} \in \mathrm{Sp}(V)$ in such a way that \bar{c} is the identity on $\langle e_1, e_{-1} \rangle$ and induces Γ on $\langle e_1, e_{-1} \rangle^\perp$.
8. Use the algorithm for (A) to write an SLP from $\bar{\mathcal{T}}$ to \bar{c} .
9. Evaluate the SLP in Step 8 from \mathcal{T} to obtain an element $c^* \in G$.
10. If $c^* = c$ set $a := 1$; if $c^* \neq c$, set $a := z$.
11. Write SLPs from Δ to each of the elements u, v .
12. Use these to construct an SLP to $g = c^* a h^j v^{-1} l^{-1} u^{-1} b^{-1}$.
13. Return the SLP in Step 12.

Commentary. The correctness of Steps 1 through 8 is clear. In Step 9, the elements c and c^* induce identical transformations on Q/T and the identity on T . Thus they can differ only in the odd characteristic case, and then only by the generator, z , of $Z(G)$. The matrix Γ in Step 6 is computed using Lemma 6.4. The SLPs in Step 11 are constructed using Lemma 6.3.

In the remainder of this section, we will complete the proof of Theorem 1.2 by establishing the following result.

Proposition 6.1. *SpSLP is a deterministic algorithm that writes an SLP of length $O(d^2 \log q)$ from \mathcal{T} to g . The complexity of the algorithm is $O(\chi d^2 + \mu d^2 \log q)$.*

Let $m = d/2$ be the Witt index of the symplectic space V , and let I be the indexing list $2, \dots, m, -2, \dots, -m$. Thus $\langle e_i : i \in I \rangle = \langle e_1, e_{-1} \rangle^\perp$. For $i \in I$, let $u_i \in \Delta \subset \mathcal{T}$ correspond to the short root element $r(e_i, 0) \in \bar{\Delta}$. Let B be the (ordered) basis $\{r_i T : i \in I\}$ of Q/T . We state the next result just for the case $p > 2$, and comment on the characteristic 2 case afterward.

Lemma 6.2. *Given any $u \in Q$, in $O(\chi d)$ -time one can obtain an \mathbb{F}_q -vector $(\alpha_i : i \in I)$ representing uT relative to B .*

Proof. If $\bar{u} = \Psi^{-1}(u) \in Q(x)$, then there exist scalars α_i, λ_i ($i \in I$) such that $\bar{u} = \prod_{i \in I} r(\alpha_i e_i, \lambda_i)$. Thus, in order to represent uT as an \mathbb{F}_q -vector relative to B , we must determine the scalars α_i . For $i \in I$, one determines $\alpha_i \in \mathbb{F}_q$ as follows: Set $t_i^* := [u_{-i}, u] \in T$; compute $\Psi_L^{-1}(t_i^*) = \begin{bmatrix} 1 & \beta \\ 0 & 1 \end{bmatrix}$; and set $\alpha_i := \beta/2$.

That these scalars are correct follows from equation (5), and the complexity is dominated by the $d - 2$ uses of the $\text{SL}(2, q)$ -oracle. \square

Lemma 6.2 is the only algorithmic task in this section that must be modified in characteristic 2. Once again, the fact that Q is abelian renders the commutator approach useless. In this case we proceed along the same lines as [7, Lemma 4.1]; the resulting algorithm, while more involved, has the same asymptotic complexity.

Lemma 6.3. *Given any $u \in Q$, in $O(\chi d + \mu \log q)$ -time one can write an SLP of length $O(d \log q)$ from Δ to u .*

Proof. Applying Lemma 6.2 one obtains a vector $(\alpha_i : i \in I) \in \mathbb{F}_q^{d-2}$ representing uT relative to B . Fix $i \in I$. Use linear algebra to write α_i as a $\text{GF}(p)$ -vector relative to $1, \rho, \dots, \rho^{k-1}$. In this way, one obtains an SLP of length $O(d \log q)$ from Δ to an element $u^* \in Q$ satisfying $uT = u^*T$. Finally, set $t^* := u(u^*)^{-1} \in T$, and use the $\text{SL}(2, q)$ -oracle to write an SLP of length $O(\log q)$ from Δ to t^* . Concatenating the two SLPs gives the desired SLP to u . \square

Lemma 6.4. *Given any $c \in N_G(T)$, in $O(\chi d^2)$ -time one can compute the $(d-2) \times (d-2)$ -matrix representing the transformation induced by c on Q/T relative to B .*

Proof. We require the matrix with rows indexed by $i \in I$, where the entries on the i th row are the coordinates of the vectors representing $(u_i T)^c$ relative to B . Each such vector is determined using Lemma 6.2. \square

Proof of Proposition 6.1. The correctness of the algorithm is proved exactly as in [16, Proposition 5.18]. The complexity is dominated by Steps 6 and 9. Step 6 uses Lemma 6.4. The remaining term in the stated complexity estimate arises from the evaluation of an SLP of length $O(d^2 \log q)$ within G to construct the element c^* in Step 9. \square

7 Implementation and performance

The algorithm presented in this paper has been implemented in **GAP** and is publicly available. An implementation of the algorithm is also under development in **MAGMA**.

In this section we comment on aspects of the GAP implementation and report on its performance. All performance tests were carried in GAP version 4.4 on a 1GHz Pentium processor.

7.1 The $\text{SL}(2, q)$ -oracle

For the theoretical analysis presented in this paper it has been assumed that the three components of the $\text{SL}(2, q)$ -oracle discussed in Section 3 are all deterministic algorithms. In particular, if the input group L is indeed isomorphic to $\text{SL}(2, q)$ for the specified value of q , then the oracle returns an effective isomorphism. If, on the other hand, L is not isomorphic to $\text{SL}(2, q)$, then the oracle returns **false**.

In practice, however, an implementation of an $\text{SL}(2, q)$ -oracle is a (randomized) 1-sided Monte Carlo algorithm. Specifically, if L is not isomorphic to $\text{SL}(2, q)$, then the algorithm will return **false** (with probability 1). On the other hand, if L is isomorphic to $\text{SL}(2, q)$, with high probability the algorithm will construct a suitable effective isomorphism. In the latter case, there is a small chance that **false** will be returned.

The GAP implementation uses an $\text{SL}(2, q)$ -oracle having two methods from which to select. The first method applies to input groups L that are in fact *equal* to $\text{SL}(2, q)$; that is, it works for groups given in their natural representation. This is based on the algorithm of Conder and Leedham-Green [11]. The second method is purely black box, and is based on the algorithm of Kantor and Seress [16, Section 3.6.1]; in fact, this is a modified version of an earlier joint implementation by Seress and the author.

Thus, if L is not given in the natural representation, then the oracle defaults to the black box method. Ideally, a third method should be available that handles groups L that are given as matrix groups in the correct characteristic. The algorithm of Conder, Leedham-Green and O'Brien [12] deals with this case, and it has been implemented in MAGMA by O'Brien.

7.2 CRecogniseSp4

The performance of the GAP implementation was assessed in dimension 4 by taking concrete black box representations of $\text{Sp}(4, q)$ and $\text{PSp}(4, q)$ from the ATLAS of Finite Group Representations (<http://web.mat.bham.ac.uk/atlas/v2.0/>). The input groups were either permutation groups, or finite matrix groups in cross-characteristic. The average time for 10 runs with each input group was taken and recorded in Table 1 under the

Table 1: Performance of `CRecogniseSp4` for ATLAS representations

group	representation	<code>CRecogniseSp4</code>	<code>SpSLP</code>
PSp(4, 3)	Sym(36)	0.05	0.03
	Sym(45)	0.06	0.03
	GL(4, 4)	0.33	0.03
	GL(6, 5)	0.76	0.05
Sp(4, 3)	Sym(80)	0.06	0.03
	Sym(240)	0.08	0.04
PSp(4, 4)	Sym(85)	0.08	0.03
	Sym(120)	0.08	0.03
	GL(18, 3)	0.45	0.14
	GL(18, 17)	1.14	0.234
PSp(4, 5)	Sym(325)	0.07	0.03
	GL(12, 4)	0.25	0.06
	GL(13, 9)	0.59	0.08
PSp(4, 7)	Sym(1176)	0.14	0.04
PSp(4, 9)	Sym(1640)	0.22	0.07
PSp(4, 19)	Sym(14480)	2.23	0.53

column `CRecogniseSp4`. The entries in the column `SpSLP` represent average times to construct an SLP to a random element of the group.

7.3 `CRecogniseSp`

In order to test the performance of the implementation for arbitrary d , we constructed “pure” black box representations of symplectic groups. (There is a paucity of concrete black box representations of $\text{PSp}(d, q)$ for $d \geq 6$.) This was done in `GAP` by creating a new family of groups that only admit the basic black box operations. Using these groups as input, one can be certain that information specific to the given representation is not being used by the algorithm. Given any matrix group (or permutation group), one may convert it to a pure black group in `GAP`, essentially by making the given group “forget” its initial construction.

A test was conducted to investigate the performance of the algorithm for fixed field size

Table 2: Performance of CRecogniseSp for pure black box groups

group	CRecogniseSp	SpSLP
Sp(10, 7)	1.1	0.34
Sp(12, 7)	1.7	0.52
Sp(14, 7)	2.6	0.80
Sp(16, 7)	4.0	2.32
Sp(18, 7)	6.0	2.53
Sp(20, 7)	8.8	4.13
Sp(30, 7)	42.8	14.47

and increasing dimension. Table 2 records the running time of the algorithm for input symplectic groups $\text{Sp}(d, 7)$, converted to a pure black box group from the natural matrix copy of the group stored in **GAP**. Once again it includes average times to construct an SLP to a random element of the group.

References

- [1] L. Babai, Local expansion of vertex transitive graphs and random generation in finite groups, pp. 164-174 in *Proc. ACM Symp. on Theory of Computing* (1991).
- [2] L. Babai, W. Kantor, P. Pálffy and Á. Seress, Black box recognition of finite simple groups of Lie type by statistics of element orders, *J. Group Theory* 5 (2002), 383-401.
- [3] W. Bosma, J. Cannon and C. Playoust, The MAGMA algebra system. I. The user language, *Computational Algebra and Number Theory (London, 1993)*. *J. Symbolic Comput.* 24 (1997), no. 3-4, 235-265.
- [4] P.A. Brooksbank, Constructive recognition of classical groups in their natural representation, *J. Symbolic Comput.* 35 (2003), no. 2, 195-239.
- [5] P.A. Brooksbank, Fast constructive recognition of black box unitary groups, *LMS J. Comput. Math.* 6 (2003), 162-197.
- [6] P.A. Brooksbank and W.M. Kantor, On constructive recognition of a black box $\text{PSL}(d, q)$, pp. 95-111 in: *Groups and Computation III (Columbus, OH, 1999)*, Ohio

- State Univ. Math. Res. Inst. Publ. vol. 8, eds. W. M. Kantor and Á. Seress (Walter de Gruyter, Berlin-New York, 2001).
- [7] P.A. Brooksbank and W.M. Kantor, Fast constructive recognition of black box orthogonal groups, *J. Algebra* 300 (2006), no. 1, 256-288.
- [8] J. Cannon and D.F. Holt, Computing maximal subgroups of finite groups, *J. Symbolic Comput.* 37 (2004), no. 5, 589-609.
- [9] F. Celler, Matrixgruppenalgorithmen in GAP, Ph.D. thesis, RWTH Aachen, 1997.
- [10] F. Celler, C.R. Leedham-Green, S.H. Murray, A.C. Niemeyer and E.A. O'Brien, Generating random elements of a finite group, *Comm. in Alg.* 23 (1995) 4931-4948.
- [11] M. Conder and C.R. Leedham-Green, Fast recognition of classical groups over large fields, pp. 113-121 in: *Groups and Computation III (Columbus, OH, 1999)*, Ohio State Univ. Math. Res. Inst. Publ. vol. 8, eds. W. M. Kantor and Á. Seress (Walter de Gruyter, Berlin-New York, 2001).
- [12] M. Conder, C.R. Leedham-Green and E.A. O'Brien, Constructive recognition of $\text{PSL}(2, q)$, *Trans. Amer. Math. Soc.* 358 (3) (2006) 1203-1221.
- [13] The GAP Group, GAP – Groups, Algorithms, and Programming, Version 4.4.9; 2006. (<http://www.gap-system.org>)
- [14] P.E. Holmes, S.A. Linton, E.A. O'Brien, A.J.E. Ryba and R.A. Wilson, Constructive membership in black box groups. Preprint, 2007.
- [15] W.M. Kantor and R.A. Liebler, The rank 3 permutation representations of the finite classical groups, *Trans. Amer. Math. Soc.* 271 (1982) 1-71.
- [16] W.M. Kantor and Á. Seress, Black box classical groups, *Memoirs of the Amer. Math. Soc.* 149 Number 708 (2001).
- [17] C.R. Leedham-Green and E.A. O'Brien, Constructive recognition of classical groups, preprint.
- [18] E.A. O'Brien, Towards effective algorithms for linear groups, pp. 163-190 in: *Finite geometries, groups, and computation*, Walter de Gruyter GmbH & Co. KG, Berlin, 2006.

- [19] C.W. Parker and R.A. Wilson, Recognising simplicity in black box groups. Preprint, 2007.
- [20] Á. Seress, Permutation group algorithms, Cambridge Univ. Press, 2002.
- [21] D.E. Taylor, *The geometry of the classical groups*, Heldermann, Berlin, 1992.

Department of Mathematics
Bucknell University
Lewisburg, PA 17837
United States
pbrooks@bucknell.edu

Last revised December 2007