RECOGNISING THE SUZUKI GROUPS IN THEIR NATURAL REPRESENTATIONS

HENRIK BÄÄRNHIELM

Abstract. Under the assumption of a certain conjecture, for which there exists strong experimental evidence, we produce an efficient algorithm for constructive membership testing in the Suzuki groups $\mathrm{Sz}(q)$, where $q = 2^{2m+1}$ for some $m > 0$, in their natural representations of degree 4. It is a Las Vegas algorithm with running time $\mathrm{O}(\log(q))$ field operations, and a preprocessing step with running time $\mathrm{O}(\log(q)\log\log(q))$ field operations. The latter step needs an oracle for the discrete logarithm problem in $\mathbb{F}_q$.

We also produce a recognition algorithm for $\mathrm{Sz}(q) = \langle X \rangle$. This is a Las Vegas algorithm with running time $\mathrm{O}(|X|^2)$ field operations.

Finally, we give a Las Vegas algorithm that, given $\langle X \rangle^h = \mathrm{Sz}(q)$ for some $h \in \mathrm{GL}(4, q)$, finds some g such that $\langle X \rangle^g = \mathrm{Sz}(q)$. The running time is $\mathrm{O}(\log(q)\log\log(q) + |X|)$ field operations.

Implementations of the algorithms are available for the computer system Magma.

1. Introduction

A goal of the *matrix recognition project* is to develop efficient algorithms for the study of subgroups of $\mathrm{GL}(d, q)$. The classification due to Aschbacher (see [1]) provides one framework for this, and the first aim is to develop an algorithm that finds a composition series of a matrix group given by a set of generators. It is possible to do this with a recursive algorithm, and the recursion is described in [16]. However, we still have to deal with the base cases, which are the finite simple groups.

For each base case we need to perform parts of *constructive recognition*. The simple group is given as $G = \langle X \rangle$ where $X \subseteq \mathrm{GL}(d, q)$ for some d, q and constructive recognition encompasses the following problems:

(1) The problem of *recognition* or *naming* of G, *i.e.* decide the name of G, as in the classification of the finite simple groups.
(2) The *constructive membership* problem. Given $g \in \mathrm{GL}(d, q)$, decide whether or not $g \in G$, and if so express g as a word (or `SLP`, see Section 3.2) in X.
(3) Construct an isomorphism ψ from G to a *standard copy* H of G such that $\psi(g)$ and $\psi^{-1}(h)$ can be computed efficiently for every $g \in G$ and $h \in H$. Sometimes this particular problem is what is meant by "constructive recognition".

To find a composition series using [16], we need only recognition and constructive membership, but the explicit isomorphisms to a standard copy are also very useful. Given these, many problems, including constructive membership, can be reduced to the standard copy.

This paper will consider the Suzuki groups $\mathrm{Sz}(q)$, $q = 2^{2m+1}$ for $m > 0$, which is one of the infinite families of finite simple groups. We will only consider the natural representation, which has dimension 4, and our standard copy will be $\mathrm{Sz}(q)$ defined in Section 2.

In Section 5 we solve the constructive membership problem for $\mathrm{Sz}(q)$. In Section 6 we solve the recognition problem for $\mathrm{Sz}(q)$, *i.e.* given $X \subseteq \mathrm{GL}(4, q)$ we give an algorithm that decides whether or not $\langle X \rangle = \mathrm{Sz}(q)$. In Section 7 we consider these problems for conjugates of $\mathrm{Sz}(q)$. Given $X \subseteq \mathrm{GL}(4, q)$ we give an algorithm that decides whether or not $\langle X \rangle^h = \mathrm{Sz}(q)$ for some $h \in \mathrm{GL}(4, q)$. We also give an algorithm that computes an isomorphism to $\mathrm{Sz}(q)$, by finding some g such that $\langle X \rangle^g = \mathrm{Sz}(q)$.

Other representations are dealt with in [2]. The main objective of this paper is to prove the following:

Theorem 1.1. *Assuming Conjecture 4.2, and given a random element oracle for subgroups of* $\mathrm{GL}(4, q)$ *and an oracle for the discrete logarithm problem in* $\mathbb{F}_q$*, there exists a Las Vegas algorithm that, for each* $X \subseteq \mathrm{GL}(4, q)$*, with* $q = 2^{2m+1}$ *for some* $m > 0$*, such that* $\langle X \rangle^h = \mathrm{Sz}(q)$ *for some* $h \in \mathrm{GL}(4, q)$*, finds* $g \in \mathrm{GL}(4, q)$ *such that* $\langle X \rangle^g = \mathrm{Sz}(q)$ *and solves the constructive membership problem for* $\langle X \rangle$*. The algorithm has time complexity* $\mathrm{O}(\log(q))$ *field operations and also has a preprocessing step, which only needs to be executed once for a given* X*, with time complexity* $\mathrm{O}(\log(q) \log\log(q) + |X|)$ *field operations. The discrete logarithm oracle is only needed in the preprocessing step.*

Proof. Follows from Theorem 7.5, Theorem 5.2, Theorem 5.3 and Theorem 5.4. □

In Section 8, experimental evidence for Conjecture 4.2 is shown.

In constructive membership testing for $\mathrm{Sz}(q)$, the essential problem is to find elements of even order. In this paper, this is achieved by using the fact that $\mathrm{Sz}(q)$ acts doubly transitively on a certain set $\mathcal{O} \subseteq \mathbb{P}^3(\mathbb{F}_q)$. After finding independent random elements in the stabiliser of a point, which is done by finding elements that map one point to another, it becomes easy to find elements of even order. This is because the structure of the stabiliser of a point is known, and by Proposition 5.1 we can easily find elements of even order in it.

For every cyclic subgroup C of order $q - 1$, the proportion of double cosets of C in $\mathrm{Sz}(q)$ that contain an element that maps one given point to another is high. The need to consider double cosets rather than single cosets arises from the fact that $\mathcal{O}$ contains $q^2 + 1$ points, and most double cosets have size $(q - 1)^2$. In the analogous problem for $\mathrm{SL}(2, q)$ (see [8]), which acts on a set with $q + 1$ points, single cosets of a subgroup of order $q - 1$ are used.

One can view this as a process of applying permutation group techniques on a set which is exponentially large in terms of the input. Since $\mathcal{O}$ has size $q^2 + 1$, we cannot explicitly write down all its points and still have a polynomial time algorithm, and therefore we cannot write down the elements of $\mathrm{Sz}(q)$ as permutations. However, given two points we can construct in polynomial time an element of $\mathrm{Sz}(q)$ that maps one point to the other, which is a typical permutation group technique.

Implementations of the algorithms are available in MAGMA (see [5]).

We are very grateful to the anonymous referee for the helpful advice and the large number of comments. We also acknowledge John Bray, Charles Leedham-Green, Eamonn O'Brien, Geoffrey Robinson, Maud de Visscher and Robert Wilson for their help and encouragement.

2. The simple Suzuki groups

We begin by defining our standard copy of the Suzuki group. Following [14, Chapter 11], let π be the unique automorphism of $\mathbb{F}_q$ such that $\pi^2(x) = x^2$ for every $x \in \mathbb{F}_q$, *i.e.* $\pi(x) = x^t$ where $t = 2^{m+1}$. For $a, b \in \mathbb{F}_q$ and $c \in \mathbb{F}_q^\times$, define the following matrices.

$$S(a,b) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ a & 1 & 0 & 0 \\ b & \pi(a) & 1 & 0 \\ a^2\pi(a) + ab + \pi(b) & a\pi(a) + b & a & 1 \end{bmatrix} \tag{2.1}$$

$$M(c) = \begin{bmatrix} c^{1+2^m} & 0 & 0 & 0 \\ 0 & c^{2^m} & 0 & 0 \\ 0 & 0 & c^{-2^m} & 0 \\ 0 & 0 & 0 & c^{-1-2^m} \end{bmatrix} \tag{2.2}$$

$$T = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.3}$$

By definition,

$$\mathrm{Sz}(q) = \left\langle S(a,b), M(c), T \mid a, b \in \mathbb{F}_q, c \in \mathbb{F}_q^{\times} \right\rangle. \tag{2.4}$$

If we define

$$\mathcal{F} = \{S(a,b) \mid a, b \in \mathbb{F}_q\} \tag{2.5}$$

$$\mathcal{H} = \left\{M(c) \mid c \in \mathbb{F}_q^{\times}\right\} \tag{2.6}$$

then $\mathcal{F} \leqslant \mathrm{Sz}(q)$ with $|\mathcal{F}| = q^2$ and $\mathcal{H} \cong \mathbb{F}_q^{\times}$ so that $\mathcal{H}$ is cyclic of order $q-1$. Moreover, we can write $M(c)$ as

$$M(c) = M'(\lambda) = \begin{bmatrix} \lambda^{t+1} & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & \lambda^{-1} & 0 \\ 0 & 0 & 0 & \lambda^{-t-1} \end{bmatrix} \tag{2.7}$$

where $\lambda = c^{2^m}$.

The following result follows from [14, Chapter 11].

Theorem 2.1. (1) *The order of the Suzuki group is*

$$|\mathrm{Sz}(q)| = (q^2+1)q^2(q-1). \tag{2.8}$$

(2) *For all $a, b, a', b' \in \mathbb{F}_q$ and $\lambda \in \mathbb{F}_q^{\times}$ we have:*

$$S(a,b)S(a',b') = S(a+a', b+b'+a^t a') \tag{2.9}$$

$$S(a,b)^{M(\lambda)} = S(\lambda a, \lambda^{t+1} b). \tag{2.10}$$

(3) *There exists $\mathcal{O} \subseteq \mathbb{P}^3(\mathbb{F}_q)$ on which $\mathrm{Sz}(q)$ acts faithfully and doubly transitively, such that no nontrivial element of $\mathrm{Sz}(q)$ fixes more than 2 points. This set is*

$$\mathcal{O} = \{(1:0:0:0)\} \cup \left\{(ab + \pi(a)a^2 + \pi(b) : b : a : 1) \mid a, b \in \mathbb{F}_q\right\}. \tag{2.11}$$

(4) *The stabiliser of $P_\infty = (1:0:0:0) \in \mathcal{O}$ is $\mathcal{FH}$ and if $P_0 = (0:0:0:1)$ then the stabiliser of (P_∞, P_0) is $\mathcal{H}$.*

(5) $\mathrm{Z}(\mathcal{F}) = \{S(0,b) \mid b \in \mathbb{F}_q\}$ *and $\mathcal{FH}$ is a Frobenius group with Frobenius kernel $\mathcal{F}$.*

(6) *The number of elements of order $q-1$ is $\phi(q-1)q^2(q^2+1)/2$, where ϕ is the Euler totient function.*

(7) *Let $g \in G = \mathrm{Sz}(q)$. Then for every $x \in G$, $\mathrm{C}_G(g) \cap \mathrm{C}_G(g)^x = \langle 1 \rangle$ if $\mathrm{C}_G(g) \neq \mathrm{C}_G(g)^x$.*

(8) $\mathrm{Sz}(q)$ *has cyclic Hall subgroups U_1 and U_2 of orders $q \pm t + 1$.*

From [14, Chapter 11, Remark 3.12] we also immediately obtain the following result.

Theorem 2.2. *A maximal subgroup of $G = \mathrm{Sz}(q)$ is conjugate to one of the following subgroups.*

(1) *The point stabiliser $\mathcal{FH}$.*
(2) *The normaliser $\mathrm{N}_G(\mathcal{H})$, which is dihedral of order $2(q-1)$.*
(3) *The normalisers $\mathcal{B}_i = \mathrm{N}_G(U_i)$ for $i = 1, 2$. These satisfy $\mathcal{B}_i = \langle U_i, t_i \rangle$ where $u^{t_i} = u^q$ for every $u \in U_i$ and $[\mathcal{B}_i : U_i] = 4$.*
(4) $\mathrm{Sz}(s)$ *where q is a power of s.*

If G is a group acting on a set $\mathcal{O}$ and $P \in \mathcal{O}$, let $G_P \leqslant G$ denote the stabiliser of P in G.

Let $\mathrm{Sp}(4, q)$ denote the standard copy of the symplectic group, preserving the following symplectic form:

$$J = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \tag{2.12}$$

From [18] and [25, Chapter 3], we know that the elements of $\mathrm{Sz}(q)$ are precisely the fixed points of an automorphism Ψ of $\mathrm{Sp}(4, q)$; from [25, Chapter 3], computing $\Psi(g)$ for some $g \in \mathrm{Sp}(4, q)$ amounts to taking a submatrix of the exterior square of g and then replacing each matrix entry x by x^{2^m}. Moreover, Ψ is defined on $\mathrm{Sp}(4, F)$ for $F \geqslant \mathbb{F}_q$.

If V is an FG-module for some group G and field F, with action $f : FG \times V \to V$, and if Φ is an automorphism of G, denote by V^{Φ} the FG-module which has the same elements as V and where the action is given by $(g, v) \mapsto f(\Phi(g), v)$ for $g \in G$ and $v \in V^{\Phi}$, extended to FG by linearity.

Lemma 2.3. *Let $G \leqslant \mathrm{Sp}(4, q)$ have natural module V and assume that V is absolutely irreducible. Then $G^h \leqslant \mathrm{Sz}(q)$ for some $h \in \mathrm{GL}(4, q)$ if and only if $V \cong V^{\Psi}$.*

Proof. Assume $G^h \leqslant \mathrm{Sz}(q)$. Both G and $\mathrm{Sz}(q)$ preserve the form (2.12), and this form is unique up to a scalar multiple, since V is absolutely irreducible. Therefore $hJh^T = \lambda J$ for some $\lambda \in \mathbb{F}_q^{\times}$. But if $\mu = \sqrt{\lambda^{-1}}$ then $(\mu h) J (\mu h)^T = J$, so that $\mu h \in \mathrm{Sp}(4, q)$. Moreover, $G^h = G^{\mu h}$, and hence we may assume that $h \in \mathrm{Sp}(4, q)$. Let $x = h\Psi(h^{-1})$ and observe that for each $g \in G$, $\Psi(g^h) = g^h$. It follows that

$$g^x = \Psi(h) g^h \Psi(h^{-1}) = \Psi(h g^h h^{-1}) = \Psi(g) \tag{2.13}$$

so $V \cong V^{\Psi}$.

Conversely, assume that $V \cong V^{\Psi}$. Then there is some $h \in \mathrm{GL}(4, q)$ such that for each $g \in G$ we have $g^h = \Psi(g)$. As above, since both G and $\Psi(G)$ preserve the form (2.12), we may assume that $h \in \mathrm{Sp}(4, q)$.

Let K be the algebraic closure of $\mathbb{F}_q$. The Steinberg-Lang Theorem (see [22]) asserts that there exists $x \in \mathrm{Sp}(4, K)$ such that $h = x^{-1}\Psi(x)$. It follows that

$$\Psi(g^{x^{-1}}) = \Psi(g)^{h^{-1}x^{-1}} = g^{x^{-1}} \tag{2.14}$$

so that $G^{x^{-1}} \leqslant \mathrm{Sz}(q)$. Thus G is conjugate in $\mathrm{GL}(4, K)$ to a subgroup S of $\mathrm{Sz}(q)$, and it follows from [10, Theorem 29.7], that G is conjugate to S in $\mathrm{GL}(4, q)$. □

Lemma 2.4. *If $H \leqslant G = \mathrm{Sz}(q)$ is a cyclic group of order $q-1$ and $g \in G \setminus \mathrm{N}_G(H)$ then $|HgH| = (q-1)^2$.*

Proof. Since $|H| = q - 1$ it is enough to show that $H \cap H^g = \langle 1 \rangle$. By [14, Chapter 11], H is conjugate to $\mathcal{H}$ and distinct conjugates of $\mathcal{H}$ intersect trivially. □

Lemma 2.5. *If $g \in G = \mathrm{Sz}(q)$ is uniformly random, then*

$$\Pr[|g| = q-1] = \frac{\phi(q-1)}{2(q-1)} > \frac{1}{12 \log\log(q)} \tag{2.15}$$

and hence we expect to obtain an element of order $q-1$ in $\mathrm{O}(\log\log q)$ random selections.

Proof. The first equality follows immediately from Theorem 2.1. The inequality follows from [17, Section II.8].

Now let $\varepsilon = 1/(12 \log\log(q))$ and $\delta = \mathrm{e}^{-k}$ for some $k \in \mathbb{N}$. If we take uniformly random elements from G, then the probability that we have not found an element of order $q-1$ after $\lceil \log\delta / \log(1-\varepsilon) \rceil$ consecutive tries is at most δ, and

$$\frac{\log\delta}{\log(1-\varepsilon)} \approx \frac{k}{\varepsilon} \tag{2.16}$$

which is $\mathrm{O}(\log\log(q))$, so the statement follows. □

Lemma 2.6. *The number of elements of $G = \mathrm{Sz}(q)$ that fix at least one point of $\mathcal{O}$ is $q^2(q-1)(q^2+q+2)/2$.*

Proof. By [14, Chapter 11], if $g \in G$ fixes exactly one point, then g is in a conjugate of $\mathcal{F}$ and if g fixes two points then g is in a conjugate of $\mathcal{H}$. This implies that there are $(|\mathcal{F}|-1)\,|\mathcal{O}|$ elements that fix exactly one point. Similarly, there are $\binom{|\mathcal{O}|}{2}(|\mathcal{H}|-1)$ elements that fix exactly two points.

Thus the number of elements that fix at least one point is

$$1 + (|\mathcal{F}|-1)\,|\mathcal{O}| + \binom{|\mathcal{O}|}{2}(|\mathcal{H}|-1) = \frac{q^2(q-1)(q^2+q+2)}{2}. \tag{2.17}$$

□

Lemma 2.7. *Elements of odd order in $\mathrm{Sz}(q)$ that have the same trace are conjugate.*

Proof. From [23], the number of conjugacy classes of non-identity elements of odd order is $q-1$, and all elements of even order have trace 0. Observe that

$$S(0,b)T = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & b \\ 1 & 0 & b & b^t \end{bmatrix}. \tag{2.18}$$

Since b can be any element of $\mathbb{F}_q$, so can $\mathrm{Tr}\,(S(0,b)T)$, and this also implies that $S(0,b)T$ has odd order when $b \neq 0$. Therefore there are $q-1$ possible traces for non-identity elements of odd order, and elements with different trace must be non-conjugate, so all conjugacy classes must have different traces. □

3. Preliminaries

We will now briefly discuss some general concepts that are needed later.

3.1. **Complexity.** We shall be concerned with the time complexity of the algorithms involved, where the basic operations are the field operations, and not the bit operations. In our case, the matrix dimension will always be 4, so all simple arithmetic with matrices can be done using $\mathrm{O}(1)$ field operations, and raising a matrix to the $\mathrm{O}(q)$ power can be done using $\mathrm{O}(\log q)$ field operations using the standard method of repeated squaring. We shall also assume an oracle for the discrete logarithm problem for $\mathbb{F}_q$, so that this can be solved using $\mathrm{O}(1)$ field operations.

We will need to find an element of order $q-1$. The order can be computed using the algorithm of [6]. To obtain the *precise* order, this algorithm requires a factorisation of $q-1$, otherwise it might return a multiple of the correct order.

However, it suffices for our purposes to learn a *pseudo-order* of the element, which is a multiple of its order, since it will suffice to find a nontrivial element of order dividing $q-1$. Hence we avoid the requirement to factorise $q-1$. The algorithm of [6] can also be used to obtain the pseudo-order, and for this it has time complexity $\mathrm{O}(\log(q)\log\log(q))$ field operations.

3.2. **Straight line programs.** For constructive membership testing, we want to express an element of a group $G = \langle X \rangle$ as a word in X. Actually, it should be a *straight line program*, abbreviated to `SLP`. If we express the elements as words, the length of the words might be too large, requiring exponential space complexity.

An `SLP` is a data structure for words, which ensures that subwords occurring multiple times are computed only once. Formally, given a set of generators X, an `SLP` is a sequence $(s_1, s_2, \dots, s_n)$ where each s_i represents one of the following

- an $x \in X$
- a product $s_j s_k$, where $j, k < i$
- a power s_j^n where $j < i$ and $n \in \mathbb{Z}$
- a conjugate $s_j^{s_k}$ where $j, k < i$

so s_i is either a pointer into X, a pair of pointers to earlier elements of the sequence, or a pointer to an earlier element and an integer.

Thus to construct an `SLP` for a word, one starts by listing pointers to the generators of X, and then builds up the word. To evaluate the `SLP`, go through the sequence and perform the specified operations. Since we use pointers to the elements of X, we can immediately evaluate the `SLP` on another set Y of the same size as X, by just changing the pointers so that they point to elements of Y.

3.3. **Random elements.** Our analysis assumes that we can construct uniformly distributed random elements of a group G defined by a generating set X. The polynomial time algorithm of [3] produces nearly uniformly distributed random elements; an alternative polynomial time algorithm is the *product replacement* algorithm of [7]. We will assume that we have a random element oracle, which produces a uniformly random element using $\mathrm{O}(1)$ field operations, and automatically gives it as an `SLP` in X.

An important issue is the length of the `SLP`s that are computed. The length of the `SLP`s must be polynomial, otherwise it would not be polynomial time to evaluate them. We assume that `SLP`s of random elements have length $\mathrm{O}(1)$.

3.4. **Las Vegas algorithms.** All the algorithms we consider are probabilistic of the type known as *Las Vegas algorithms*. This type of algorithm is discussed in [24, Section 25.8], [20, Section 1.3] and [12, Section 3.2.1]. In short it is a probabilistic algorithm with an input parameter ε that either returns `failure`, with probability at most ε, or otherwise returns a correct result. The time complexity naturally depends on ε.

We present Las Vegas algorithms as probabilistic algorithms that either return a correct result, with probability bounded below by $1/p(n)$ for some polynomial $p(n)$ in the size n of the input, or otherwise return `failure`. By enclosing such an algorithm in a loop that iterates $\lceil \log \varepsilon / \log(1 - 1/p(n)) \rceil$ times, we obtain an algorithm that returns `failure` with probability at most ε, and hence is a Las Vegas algorithm in the above sense. Clearly if the enclosed algorithm is polynomial time, the Las Vegas algorithm is polynomial time.

One can also enclose the algorithm in a loop that iterates until the algorithm returns a correct result, thus obtaining a probabilistic time complexity, and the expected number of iterations is then $\mathrm{O}(p(n))$.

4. Computing an element of a stabiliser

As explained in the introduction, in constructive membership testing for $\mathrm{Sz}(q)$ the essential problem is to find an element of the stabiliser of a given point $P \in \mathcal{O}$, expressed as an `SLP` in our given generators X of $G = \mathrm{Sz}(q)$. The idea is to map P to $Q \neq P$ by a random $g_1 \in G$, and then compute $g_2 \in G$ such that $Pg_2 = Q$, so that $g_1 g_2^{-1} \in G_P$.

Thus the problem is to find an element that maps P to Q, and the idea is to look for it in double cosets of cyclic subgroups of order $q-1$. We first give an overview of the method.

Begin by selecting random $a, h \in G$ such that a has pseudo-order $q-1$, and consider the equation

$$Pa^j h a^i = Q \tag{4.1}$$

in the two indeterminates i, j. If we can solve this equation for i and j, thus obtaining positive integers k, l such that $1 \leqslant k, l \leqslant q-1$ and $Pa^l h a^k = Q$, then we have an element that maps P to Q.

Since a has order dividing $q-1$, by [14, Chapter 11], a is conjugate to a matrix $M'(\lambda)$ for some $\lambda \in \mathbb{F}_q^{\times}$. This implies that we can diagonalise a and obtain a matrix $x \in \mathrm{GL}(4,q)$ such that $M'(\lambda)^x = a$. It follows that if we define $P' = Px^{-1}$, $Q' = Qx^{-1}$ and $g = h^{x^{-1}}$ then (4.1) is equivalent to

$$P' M'(\lambda)^j g M'(\lambda)^i = Q'. \tag{4.2}$$

Now change indeterminates to α and β by letting $\alpha = \lambda^j$ and $\beta = \lambda^i$, so that we obtain the following equation:

$$P' M'(\alpha) g M'(\beta) = Q'. \tag{4.3}$$

This determines four equations in α and β, and in Section 4.1 we will describe how to find solutions for them. A solution $(\gamma, \delta) \in \mathbb{F}_q^{\times} \times \mathbb{F}_q^{\times}$ determines $M'(\gamma), M'(\delta) \in \mathcal{H}$, and hence also $c, d \in H = \mathcal{H}^x$.

If $|a| = q-1$ then $\langle a \rangle = H$, so that there exists positive integers k and l as above with $a^l = c$ and $a^k = d$, and these integers can be found by computing discrete logarithms, since we also have $\lambda^l = \gamma$ and $\lambda^k = \delta$. Hence we obtain a solution to (4.1) from the solution to (4.3). If $|a|$ is a proper divisor of $q-1$, then it might happen that $c \notin \langle a \rangle$ or $d \notin \langle a \rangle$, but by Lemma 2.5 we know that this is unlikely.

Thus the overall algorithm is as in Algorithm 1. We show the time complexity of the algorithm in Section 4.2 and prove that it is correct in Section 4.3.

4.1. **Solving equation** (4.3).

We will now show how to obtain the solutions of (4.3). It might happen that there are no solutions, in which case the method described here will detect this and return with failure.

By letting $P' = (q_1 : q_2 : q_3 : q_4)$, $Q' = (r_1 : r_2 : r_3 : r_4)$ and $g = [g_{i,j}]$, we can write out (4.3) and obtain

$$\begin{aligned}
(q_1 g_{1,1} \alpha^{t+1} + q_2 g_{2,1} \alpha + q_3 g_{3,1} \alpha^{-1} + q_4 g_{4,1} \alpha^{-t-1}) \beta^{t+1} &= C r_1 \\
(q_1 g_{1,2} \alpha^{t+1} + q_2 g_{2,2} \alpha + q_3 g_{3,2} \alpha^{-1} + q_4 g_{4,2} \alpha^{-t-1}) \beta &= C r_2 \\
(q_1 g_{1,3} \alpha^{t+1} + q_2 g_{2,3} \alpha + q_3 g_{3,3} \alpha^{-1} + q_4 g_{4,3} \alpha^{-t-1}) \beta^{-1} &= C r_3 \\
(q_1 g_{1,4} \alpha^{t+1} + q_2 g_{2,4} \alpha + q_3 g_{3,4} \alpha^{-1} + q_4 g_{4,4} \alpha^{-t-1}) \beta^{-t-1} &= C r_4
\end{aligned} \tag{4.4}$$

for some constant $C \in \mathbb{F}_q$. Henceforth, we assume that $r_i \neq 0$ for $i = 1, \ldots, 4$, since this is the difficult case, and also extremely likely when q is large, as can be seen from Proposition 4.1. A method similar to the one described in this section will solve (4.3) when some $r_i = 0$ and Algorithm 1 does not assume that all $r_i \neq 0$.

Algorithm 1: `FindMappingElement`

Data: Generating set X for $G = \mathrm{Sz}(q)$ and points $P \neq Q \in \mathcal{O}$
Result: An element g of G, written as an `SLP` in X, such that $Pg = Q$
/* Assumes the existence of a function `SolveEquation` that solves (4.3), if possible. Also, assumes that the function `Random` returns an element as an `SLP` in X, and that `DiscreteLog` returns a positive integer if a discrete logarithm exists and 0 otherwise. */

begin
  $h :=$ `Random`(G)
  /* Find random element a of pseudo-order $q-1$ */
  $a :=$ `Random`(G)
  if $|a| \mid q-1$ **then**
    $(M'(\lambda), x) :=$ `Diagonalise`(a)
    /* Now $M'(\lambda)^x = a$ */
    if *SolveEquation*$(h^{x^{-1}}, Px^{-1}, Qx^{-1})$ **then**
      Let (γ, δ) be a solution.
      $l :=$ `DiscreteLog`(λ, γ)
      $k :=$ `DiscreteLog`(λ, δ)
      if $k > 0$ ***and*** $l > 0$ **then**
        return $a^l h a^k$
      end
    end
  end
  return *fail*
end

Proposition 4.1. *If $P' = (p_1 : p_2 : p_3 : p_4) \in \mathcal{O}^x$ is uniformly random, where $\mathcal{O}^x = \{Px \mid P \in \mathcal{O}\}$ for some $x \in \mathrm{GL}(4, q)$, then*

$$\Pr[p_i \neq 0 \mid i = 1, \dots, 4] \geqslant (1 - \frac{\sqrt{2q}}{q})^4. \tag{4.5}$$

Proof. Let $P' = Px$ and $x = [x_{i,j}]$. If $P = (1 : 0 : 0 : 0)$ then $P' = (x_{1,1} : x_{1,2} : x_{1,3} : x_{1,4})$ so clearly

$$\begin{aligned}\Pr[p_i = 0 \mid \text{some } i] \leqslant \frac{1}{|\mathcal{O}|} + (1 - \frac{1}{|\mathcal{O}|})\\ (1 - \Pr[(a^{t+2} + b^t + ab)x_{1,1} + x_{2,1}b + x_{3,1}a + x_{4,1} \neq 0 \mid a \neq 0, b \neq 0]^4).\end{aligned} \tag{4.6}$$

Now it follows that

$$\begin{aligned}&\Pr[(a^{t+2} + b^t + ab)x_{1,1} + x_{2,1}b + x_{3,1}a + x_{4,1} = 0 \mid a \neq 0, b \neq 0] = \\ &= \sum_{k \in \mathbb{F}_q^\times} \Pr[(k^{t+2} + b^t + kb)x_{1,1} + x_{2,1}b + x_{3,1}k + x_{4,1} = 0 \mid a = k, b \neq 0] \Pr[a = k] \leqslant \frac{t}{q}\end{aligned} \tag{4.7}$$

since in a field a polynomial of degree t has at most t roots. The result follows by observing that $t = \sqrt{2q}$. □

For convenience, we denote the expressions in the parentheses at the left hand sides of (4.4) as K, L, M and N respectively. Then if we let $C = L\beta r_2^{-1}$ we obtain three equations

$$\begin{aligned} K\beta^t &= r_1 r_2^{-1} L \\ M\beta^{-2} &= r_3 r_2^{-1} L \\ N\beta^{-t-2} &= r_4 r_2^{-1} L \end{aligned} \tag{4.8}$$

and in particular β is a function of α, since

$$\beta = \sqrt{L^{-1} M r_3^{-1} r_2}. \tag{4.9}$$

By substituting the first two equations into the third in (4.8) we obtain

$$NKr_2r_3 = r_1r_4ML \tag{4.10}$$

and by raising the first equation to the t-th power and substituting into the second, we obtain

$$r_1 r_3^{t/2} L^{1+t/2} = r_2^{1+t/2} M^{t/2} K. \tag{4.11}$$

If instead we let $C = M\beta^{-1}r_3^{-1}$ and proceed similarly, we obtain two more equations

$$N^t L r_3^{t+1} = M^{t+1} r_2 r_4^t \tag{4.12}$$

$$N L^{t/2} r_3^{1+t/2} = M^{1+t/2} r_4 r_2^{t/2}. \tag{4.13}$$

Now (4.10), (4.11), (4.12) and (4.13) are equations in α only, and by multiplying them by suitable powers of α, they can be turned into polynomial equations such that α only occurs to the powers ti for $i = 1, \dots, 4$ and to lower powers that are independent of t. The suitable powers of α are $2t+2$, $t + t/2 + 2$, $2t + 3$ and $2t + t/2 + 2$, respectively.

Thus we obtain the following four equations.

$$\begin{aligned} \alpha^{4t}c_1 + \alpha^{3t}c_2 + \alpha^{2t}c_3 + \alpha^t c_4 &= d_1 \\ \alpha^{4t}c_5 + \alpha^{3t}c_6 + \alpha^{2t}c_7 + \alpha^t c_8 &= d_2 \\ \alpha^{4t}c_9 + \alpha^{3t}c_{10} + \alpha^{2t}c_{11} + \alpha^t c_{12} &= d_3 \\ \alpha^{4t}c_{13} + \alpha^{3t}c_{14} + \alpha^{2t}c_{15} + \alpha^t c_{16} &= d_4 \end{aligned} \tag{4.14}$$

The c_i and d_j are polynomials in α with degree independent of t, for $i = 1, \dots, 16$ and $j = 1, \dots, 4$ respectively, so (4.14) can be considered a linear system in the variables α^{nt} for $n = 1, \dots, 4$, with coefficients c_i and d_j. Now the aim is to obtain a single polynomial in α of bounded degree. For this we need the following conjecture.

Conjecture 4.2. *For every $P' = Px^{-1}, Q' = Qx^{-1}, g = h^{x^{-1}}$ where $P, Q \in \mathcal{O}$, $h \in G$ and $x \in \mathrm{GL}(4, q)$, if we regard (4.14) as simultaneous linear equations in the variables α^{nt} for $n = 1, \dots, 4$, over the polynomial ring $\mathbb{F}_q[\alpha]$, then it has non-zero determinant.*

In other words, the determinant of the coefficients c_i is not the zero polynomial. We comment on the validity of Conjecture 4.2 in Section 8.

Lemma 4.3. *Given P', Q' and g as in Conjecture 4.2 and assuming Conjecture 4.2, there exists a univariate polynomial $f(\alpha) \in \mathbb{F}_q[\alpha]$ of degree at most 60, such that for every $(\gamma, \delta) \in \mathbb{F}_q^\times \times \mathbb{F}_q^\times$ that is a solution for (α, β) in (4.3) we have $f(\gamma) = 0$.*

Proof. So far in this section we have shown that if we can solve (4.14) we can also solve (4.3). From the four equations of (4.14) we can eliminate α^t. We can solve for α^{4t} from the fourth equation, and substitute into the third, thus obtaining a rational expression with no occurrence of α^{4t}. Continuing this way and substituting into the other equations, we obtain an expression for α^t in terms of the c_i and the d_i only. This can be substituted into any of the equations of (4.14), where α^{nt} for $n = 1, \dots, 4$ is obtained by powering up the expression for α^t. Thus we obtain a rational expression $f_1(\alpha)$ of degree independent of t. We now take $f(\alpha)$ to be the numerator of f_1.

In other words, we think of the α^{nt} as independent variables and of (4.14) as a linear system over these variables, with coefficients in $\mathbb{F}_q[\alpha]$. By Conjecture 4.2 we can solve this linear system.

Two possible problems can occur: f is identically zero or some of the denominators of the expressions for α^{nt}, $n = 1, \dots, 4$ turn out to be 0. However, Conjecture 4.2 rules out these possibilities. By Cramer's rule, the expression for α^t is a rational expression where the numerator is a determinant, so it consists of sums of products of c_i and d_j. Each product consists of three c_i and one d_j. By considering the calculations leading up to (4.14), it is clear that each of the products has degree at most 15. Therefore the expression for α^{4t} and hence also $f(\alpha)$ has degree at most 60.

We have only done elementary algebra to obtain $f(\alpha)$ from (4.14), and it is clear that (4.14) was obtained from (4.4) by elementary means only. Hence all solutions (γ, δ) to (4.4) must also satisfy $f(\gamma) = 0$, although there may not be any such solutions, and $f(\alpha)$ may also have other zeros. □

Corollary 4.4. *Assuming Conjecture 4.2, there exists a Las Vegas algorithm that, given P', Q' and g as in Conjecture 4.2, finds all $(\gamma, \delta) \in \mathbb{F}_q^{\times} \times \mathbb{F}_q^{\times}$ that are solutions of* (4.3)*. The algorithm has time complexity* $\mathrm{O}(\log q)$ *field operations.*

Proof. Let $f(\alpha)$ be the polynomial constructed in Lemma 4.3. To find all solutions to (4.3), we find the zeros γ of $f(\alpha)$, compute the corresponding δ for each zero γ using (4.9), and check which pairs (γ, δ) satisfy (4.4). These pairs must be all solutions of (4.3).

The only work needed is simple matrix arithmetic, finding the roots of a polynomial of bounded degree over $\mathbb{F}_q$, and raising matrices to the power t, where $t \in \mathrm{O}(q)$. Hence the time complexity is $\mathrm{O}(\log q)$ field operations and the algorithm is Las Vegas since by [24, Corollary 14.16] the algorithm for finding the roots of $f(\alpha)$ is Las Vegas with this time complexity. □

By following the procedure outlined in Lemma 4.3, it is straightforward to obtain an expression for $f(\alpha)$, where the coefficients are expressions in the entries of g, P' and Q', but we will not display it here, since it would take up too much space.

4.2. **Complexity.**

Theorem 4.5. *Given an oracle for the discrete logarithm problem in $\mathbb{F}_q$ and a random element oracle for G, the time complexity of Algorithm 1 is* $\mathrm{O}(\log(q) \log\log(q))$ *field operations.*

Proof. Diagonalising a matrix uses $\mathrm{O}(\log q)$ field operations, since it involves finding the eigenvalues, *i.e.* finding the roots of a polynomial of constant degree over $\mathbb{F}_q$, see [24, Corollary 14.16].

Computing the pseudo-order of a matrix uses $\mathrm{O}(\log(q) \log\log(q))$ field operations, if we use the algorithm described in [6]. From Corollary 4.4, it follows that line 1 uses $\mathrm{O}(\log q)$ field operations.

Finally, line 1 uses $\mathrm{O}(\log q)$ field operations, since the exponents are $\mathrm{O}(q)$. We conclude that Algorithm 1 uses $\mathrm{O}(\log(q) \log\log(q))$ field operations. □

4.3. **Correctness.** There are two issues when considering the correctness of Algorithm 1. Using the notation in the algorithm, we have to show that (4.3) has a solution with high probability, and that the integers k and l are positive with high probability.

The algorithm in Corollary 4.4 tries to find an element in the double coset $\mathcal{H}g\mathcal{H}$, where $g = h^{x^{-1}}$, and we will see that this succeeds with high probability when $g \notin \mathrm{N}_G(\mathcal{H})$, which is very likely.

If the element a has order precisely $q-1$, then from the discussion at the beginning of Section 4, we know that the integers k and l will be positive. By Lemma 2.5 we know that it is likely that a has order precisely $q-1$ rather than just a divisor of $q-1$.

Hence it follows that Algorithm 1 has high probability of success. We formalise this argument in the following results.

Lemma 4.6. *Assume Conjecture 4.2. Let $G = \mathrm{Sz}(q)$ and let $P \in \mathcal{O}$ and $a, h \in G$ be given, such that $|a| = q-1$. Let $Q \in \mathcal{O}$ be uniformly random. If $h \notin \mathrm{N}_G(\langle a \rangle)$, then*

$$\frac{(q-1)^2}{(q^2+1)\deg f} \leqslant \Pr[Q \in P \langle a \rangle h \langle a \rangle] \leqslant \frac{(q-1)^2}{q^2+1} \tag{4.15}$$

where $f(\alpha)$ is the polynomial constructed in Lemma 4.3. If instead $h \in \mathrm{N}_G(\langle a \rangle)$ then

$$\Pr[Q \in P \langle a \rangle h \langle a \rangle] = \frac{(q-1)(q^2-1)+2}{(q^2+1)^2}. \tag{4.16}$$

Proof. If $h \notin \mathrm{N}_G(\langle a \rangle)$ then by Lemma 2.4, $|\langle a \rangle h \langle a \rangle| = (q-1)^2$, and hence $|P \langle a \rangle h \langle a \rangle| \leqslant (q-1)^2$.

On the other hand, for every $Q \in \mathcal{O}$ we have

$$|\{(k_1, k_2) \mid k_1, k_2 \in \langle a \rangle, Pk_1hk_2 = Q\}| \leqslant \deg f \tag{4.17}$$

since this is the equation we consider in Section 4.1, and from Lemma 4.3 we know that all solutions must be roots of f. Thus $|P \langle a \rangle h \langle a \rangle| \geqslant |\langle a \rangle h \langle a \rangle| / \deg f$. Since Q is uniformly random from $\mathcal{O}$, and $|\mathcal{O}| = q^2+1$, the result follows.

If $h \in \mathrm{N}_G(\langle a \rangle)$ then $\langle a \rangle h \langle a \rangle = h \langle a \rangle$ and $|Ph \langle a \rangle| = |\langle a \rangle|$ if $\langle a \rangle$ does not fix Ph. By [14, Chapter 11], the number of cyclic subgroups of order $q-1$ is $\binom{|\mathcal{O}|}{2}$ and $|\mathcal{O}|-1$ such subgroups fix Ph. Moreover, if $\langle a \rangle$ fixes Ph then $Ph \langle a \rangle = \{Ph\}$. Thus

$$\begin{aligned}\Pr[Q \in P \langle a \rangle h \langle a \rangle] &= \Pr[Q \in Ph \langle a \rangle] \Pr[Pha \neq Ph] + \\ &+ \Pr[Q = Ph] \Pr[Pha = Ph] = \frac{|Ph \langle a \rangle|}{|\mathcal{O}|}\left(1 - \frac{|\mathcal{O}|-1}{\binom{|\mathcal{O}|}{2}}\right) + \frac{1}{|\mathcal{O}|}\frac{|\mathcal{O}|-1}{\binom{|\mathcal{O}|}{2}}\end{aligned} \tag{4.18}$$

and the result follows. $\square$

Theorem 4.7. *Assuming Conjecture 4.2 and given a random element oracle for G and an oracle for the discrete logarithm problem in $\mathbb{F}_q$, Algorithm 1 is a Las Vegas algorithm that with probability s returns an element mapping P to Q, where*

$$s > \frac{1}{12 \log\log(q) \deg f} + \mathrm{O}(1/q) \tag{4.19}$$

Proof. We use the notation from the algorithm. Let $g = h^{x^{-1}}$, $H = \mathcal{H}^x$, $P' = Px^{-1}$ and $Q' = Qx^{-1}$. Corollary 4.4 implies that line 1 will succeed if $Q' \in P'\mathcal{H}g\mathcal{H}$. If $|a| = q-1$, then $H = \langle a \rangle$, and the previous condition is equivalent to $Q \in P \langle a \rangle h \langle a \rangle$.

Moreover, if $|a| = q-1$ then line 1 will always succeed. It might of course succeed when $|a|$ is a proper divisor of $q-1$, so it follows that s satisfies the following inequality.

$$\begin{aligned} s \geqslant \Pr[|a| = q-1](&\Pr[h \in \mathrm{N}_G(\langle a \rangle)] \Pr[Q \in P \langle a \rangle h \langle a \rangle \mid h \in \mathrm{N}_G(\langle a \rangle)] + \\ &+ \Pr[h \notin \mathrm{N}_G(\langle a \rangle)] \Pr[Q \in P \langle a \rangle h \langle a \rangle \mid h \notin \mathrm{N}_G(\langle a \rangle)]) \end{aligned} \tag{4.20}$$

Since h is uniformly random, using Theorem 2.2 we obtain

$$\Pr[h \in \mathrm{N}_G(\langle a \rangle)] = \frac{2(q-1)}{|G|} = \frac{2}{q^2(q^2+1)} \tag{4.21}$$

From Lemma 2.5 and Lemma 4.6 we obtain

$$s \geqslant \frac{\phi(q-1)}{2(q-1)} \left[\frac{(q-1)^2}{(q^2+1)\deg f} - \frac{2}{q^2(q^2+1)} \frac{(q-1)^2}{(q^2+1)} + \frac{2}{q^2(q^2+1)} \frac{2+(q-1)(q^2-1)}{(q^2+1)^2} \right] = \\ = \frac{\phi(q-1)}{2(q-1)\deg f} + \mathrm{O}(1/q) \tag{4.22}$$

and the probability of success follows from Lemma 2.5.

Clearly if a solution is returned, it is correct, so the algorithm is Las Vegas. □

Corollary 4.8. *Assuming Conjecture 4.2 and given a random element oracle for subgroups of* $\mathrm{GL}(4, q)$ *and an oracle for the discrete logarithm problem in* $\mathbb{F}_q$*, there exists a Las Vegas algorithm that, given* $X \subseteq \mathrm{GL}(4, q)$ *such that* $G = \langle X \rangle = \mathrm{Sz}(q)$ *and* $P \in \mathcal{O}$*, finds a uniformly random* $g \in G_P$*, expressed as an* `SLP` *in* X*. The algorithm has time complexity* $\mathrm{O}(\log(q) \log\log(q))$ *field operations. If* s *is as in Theorem 4.7, the probability of success is*

$$s(1 - \frac{1}{|\mathcal{O}|}) > \frac{1}{12 \log\log(q) \deg f} + \mathrm{O}(1/q). \tag{4.23}$$

Proof. We compute g as follows.

(1) Find random $x \in G$. Let $Q = Px$ and return with failure if $P = Q$.
(2) Use Algorithm 1 to find $y \in G$ such that $Qy = P$.
(3) Now $g = xy \in G_P$.

Clearly this is a Las Vegas algorithm with probability of success as stated. Moreover, the dominating term in the complexity is the call to Algorithm 1, with time complexity given by Theorem 4.5.

The element g will be expressed as an `SLP` in X, since x is random and elements from Algorithm 1 are expressed as `SLP`s.

Each call to Algorithm 1 uses independent random elements, so the double cosets under consideration are uniformly random and independent. Therefore the elements returned by Algorithm 1 must be uniformly random. This implies that g is uniformly random. □

5. Constructive membership testing

We will now give an algorithm for constructive membership testing in $\mathrm{Sz}(q)$. Given a set of generators X, such that $G = \langle X \rangle = \mathrm{Sz}(q)$, and given $g \in G$, we want to express g as an `SLP` in X. We need the following result.

Proposition 5.1. *If* $g_1, g_2 \in \mathcal{FH}$ *are uniformly random, then*

$$\Pr[|[g_1, g_2]| = 4] = 1 - \frac{1}{q-1}. \tag{5.1}$$

Proof. Let $A = \mathcal{FH}/\mathrm{Z}(\mathcal{F})$. By Theorem 2.1, $[g_1, g_2] \in \mathcal{F}$ and has order 4 if and only if $[g_1, g_2] \notin \mathrm{Z}(\mathcal{F}) \triangleleft \mathcal{FH}$. It therefore suffices to find the proportion of pairs $k_1, k_2 \in A$ such that $[k_1, k_2] = 1$.

If $k_1 = 1$ then k_2 can be any element of A, which contributes $q(q-1)$ pairs. If $1 \neq k_1 \in \mathcal{F}/\mathrm{Z}(\mathcal{F}) \cong \mathbb{F}_q$ then $\mathrm{C}_A(k_1) = \mathcal{F}/\mathrm{Z}(\mathcal{F})$, so we again obtain $q(q-1)$ pairs. Finally, if $k_1 \notin \mathcal{F}/\mathrm{Z}(\mathcal{F})$ then $|\mathrm{C}_A(k_1)| = q-1$ so we obtain $q(q-2)(q-1)$ pairs. Thus we obtain $q^2(q-1)$ pairs from a total of $|A \times A| = q^2(q-1)^2$ pairs, and the result follows. □

The algorithm for constructive membership testing has a preprocessing step and a main step. The preprocessing step consists of finding "standard generators" for $O_2(G_{P_\infty}) = \mathcal{F}$ and $O_2(G_{P_0})$. In the case of $O_2(G_{P_\infty})$ the standard generators are defined as matrices $\{S(a_i, x_i)\}_{i=1}^n \cup \{S(0, b_i)\}_{i=1}^n$ for some unspecified $x_i \in \mathbb{F}_q$, such that $\{a_1, \ldots, a_n\}$ and $\{b_1, \ldots, b_n\}$ form vector space bases of $\mathbb{F}_q$ over $\mathbb{F}_2$ (so $n = \log_2 q = 2m + 1$).

For every $a, b \in \mathbb{F}_q$, every matrix $S(a, b) \in G_{P_\infty}$ can be reduced to the identity by multiplying it by some of the standard generators of $O_2(G_{P_\infty})$, and similarly for G_{P_0}. The standard generators are therefore used in the main step to perform row operations in G_{P_∞} and G_{P_0}.

Theorem 5.2. *Assuming Conjecture 4.2 and given a random element oracle for G and an oracle for the discrete logarithm problem in $\mathbb{F}_q$, the preprocessing step is a Las Vegas algorithm that finds standard generators for $O_2(G_{P_\infty})$ and $O_2(G_{P_0})$. The preprocessing step has time complexity* $\mathrm{O}(\log(q) \log\log(q))$ *field operations. The probability of success is at least*

$$r^4 \frac{\phi(q-1)^2(q-2)^2}{(q-1)^4} > \frac{1}{2^{10}3^6(\log\log(q))^6(\deg f)^4} + \mathrm{O}(1/q) \tag{5.2}$$

where r is the success probability of the algorithm described in Corollary 4.8.

Proof. The preprocessing step is the following:

(1) Find random $a_1, a_2 \in G_{P_\infty}$ and $b_1, b_2 \in G_{P_0}$ using the algorithm described in Corollary 4.8. Let $c_1 = [a_1, a_2]$, $c_2 = [b_1, b_2]$.
(2) Determine if $|c_1| = |c_2| = 4$, if $|a_1|$ or $|a_2|$ divides $q - 1$ and if $|b_1|$ or $|b_2|$ divides $q - 1$. Return with failure if any of these turn out to be false.
(3) Let $d_1 \in \{a_1, a_2\}$ where $|d_1|$ divides $q - 1$, and let $d_2 \in \{b_1, b_2\}$ where $|d_2|$ divides $q - 1$. Let $Y_\infty = \{c_1, d_1\}$ and $Y_0 = \{c_2, d_2\}$. Diagonalise d_1 and obtain $M'(\lambda) \in G$, where $\lambda \in \mathbb{F}_q^\times$. Determine if λ lies in a proper subfield of $\mathbb{F}_q$, and if so return with failure. Do similarly for d_2.
(4) As standard generators for $O_2(G_{P_\infty})$ we now take
$$L = \bigcup_{i=1}^{2m+1} \left\{ c_1^{d_1^i}, (c_1^2)^{d_1^i} \right\} \tag{5.3}$$
and similarly we obtain U for $O_2(G_{P_0})$.

It follows from (2.9) and (2.10) that (5.3) provides the standard generators for G_{P_∞}. These are expressed as `SLP`s in X, since this is true for the elements returned from the algorithm described in Corollary 4.8.

By Corollary 4.8, the first step succeeds with probability r^4, and the random elements selected are uniformly distributed and independent. Since $G_{P_\infty} = \mathcal{F}\mathcal{H}$, the proportion of elements of order $q - 1$ in G_{P_∞} is $\phi(q-1)/(q-1)$, and similarly for G_{P_0}. Hence by Proposition 5.1, the second step succeeds with probability at least $(\phi(q-1)^2(q-2)^2)/(q-1)^4$. If $|d_1| = |d_2| = q - 1$, the third step will also succeed, since λ will not lie in a proper subfield. Hence $O_2(G_{P_\infty}) < \langle Y_\infty \rangle \leqslant G_{P_\infty}$ and $\langle Y_\infty \rangle = G_{P_\infty}$ precisely when d_1 has order $q - 1$, and similarly for Y_0.

By the remark preceding the theorem, L determines two sets of field elements $\{a_1, \ldots, a_{2m+1}\}$ and $\{b_1, \ldots, b_{2m+1}\}$. In this case each $a_i = a\lambda^i$ and $b_i = b\lambda^{i(t+1)}$, for some fixed $a, b \in \mathbb{F}_q^\times$, where λ is as in the algorithm. Since λ does not lie in a proper subfield, these sets form vector space bases of $\mathbb{F}_q$ over $\mathbb{F}_2$.

It then follows from Lemma 2.5 and Corollary 4.8 that the probability of success of the preprocessing step is as stated. Therefore the preprocessing step is a Las Vegas algorithm.

We only determine if d_1 and d_2 have order dividing $q-1$ in order to obtain a polynomial time algorithm. To determine if λ lies in a proper subfield it suffices to determine if $|\lambda| \mid 2^n - 1$ where n is a proper divisor of $2m+1$. Hence the dominating term in the complexity is the computation of random elements in the stabiliser, in the first step. The time complexity is therefore the same as for the algorithm described in Corollary 4.8. □

Now we consider the algorithm that expresses g as an `SLP` in X. It is given formally as Algorithm 2.

Algorithm 2: `ElementToSLP`

Data: Standard generators L for G_{P_∞} and U for G_{P_0}. Matrix $g \in \langle X \rangle = G$.
Result: A `SLP` for g in X.

1 **begin**
2     $r :=$ `Random`(G)
3   **if** *gr has an eigenspace $Q \in \mathcal{O}$* **then**
4   Find $z_1 \in G_{P_\infty}$ using L such that $Qz_1 = P_0$.
        /* Now $(gr)^{z_1} \in G_{P_0}$. */
5   Find $z_2 \in G_{P_0}$ using U such that $(gr)^{z_1} z_2 = M'(\lambda)$ for some $\lambda \in \mathbb{F}_q^{\times}$.
        /* Express diagonal matrix as `SLP` */
6         $x := \operatorname{Tr}(M'(\lambda))$
7   Find $h = [S(0, (x^t)^{1/4}), S(0,1)^T]$ using $L \cup U$.
        /* Now $\operatorname{Tr} h = x$. */
8   Let $P_1, P_2 \in \mathcal{O}$ be the fixed points of h.
9   Find $a \in G_{P_\infty}$ using L such that $P_1 a = P_0$.
10   Find $b \in G_{P_0}$ using U such that $(P_2 a)b = P_\infty$.
        /* Now $h^{ab} \in G_{P_\infty} \cap G_{P_0} = \mathcal{H}$, so $h^{ab} \in \{M'(\lambda)^{\pm 1}\}$. */
11   **if** $h^{ab} = M'(\lambda)$ **then**
12   Let W be an `SLP` for $(h^{ab} z_2^{-1})^{z_1^{-1}} r^{-1}$.
13   **return** W
14   **else**
15   Let W be an `SLP` for $((h^{ab})^{-1} z_2^{-1})^{z_1^{-1}} r^{-1}$.
16   **return** W
17   **end**
18   **end**
19   **return** *fail*
20 **end**

Theorem 5.3. *Given a random element oracle for G, Algorithm 2 is a Las Vegas algorithm with probability of success* $1/2 + \mathrm{O}(1/q)$.

Proof. First observe that since r is randomly chosen we obtain it as an `SLP`. On line 2 we check if gr fixes a point, and from Lemma 2.6 we see that

$$\Pr[gr \text{ fixes a point}] = \frac{q^2+q+2}{2(q^2+1)} \approx \frac{1}{2} \tag{5.4}$$

The elements found at lines 2 and 2 can be computed using row operations, so we can obtain them as `SLP`s.

The element h found at line 2 clearly has trace x, and it can be computed using row operations, so we obtain it as an SLP. From Lemma 2.7 we know that h is conjugate to $M'(\lambda)$ and therefore must fix 2 points of $\mathcal{O}$. Hence lines 2 and 2 make sense, and the elements found can again be computed using row operations and therefore we obtain them as SLPs.

The only elements in $\mathcal{H}$ that are conjugate to h are $M'(\lambda)^{\pm 1}$, so clearly h^{ab} must be one of them.

Finally, the elements that make up W were found as SLPs, and it is clear that if we evaluate W we obtain g. Hence the algorithm is Las Vegas and the theorem follows. □

5.1. **Complexity.**

Theorem 5.4. *Given a random element oracle for G, Algorithm 2 has time complexity $\mathrm{O}(\log q)$ field operations, space complexity $\mathrm{O}(\log^2 q)$ and the length of the returned* SLP *is $\mathrm{O}(\log q)$.*

Proof. From (5.3) we see that the number of standard generators is $\mathrm{O}(\log q)$, and each matrix uses $\mathrm{O}(\log q)$ space, so the space complexity of the algorithm is $\mathrm{O}(\log^2 q)$.

This also immediately implies that the row operations performed at lines 2, 2, 2, 2 and 2 use $\mathrm{O}(\log q)$ field operations.

Finding the fixed points of h, and performing the check at line 2 only amounts to considering eigenspaces, which uses $\mathrm{O}(\log q)$ field operations. Thus the time complexity of the algorithm is $\mathrm{O}(\log q)$ field operations.

The SLPs returned from Algorithm 1 have length $\mathrm{O}(1)$, and (5.3) implies that each standard generator also has length $\mathrm{O}(1)$. Hence because of our row operations, W will have length $\mathrm{O}(\log q)$. □

6. Recognition

We now discuss how to recognise $\mathrm{Sz}(q)$. We are given a set $X \subseteq \mathrm{GL}(4, q)$ and we want to decide whether or not $\langle X \rangle = \mathrm{Sz}(q)$, the group defined in (2.4).

To do this, it suffices to determine if $X \subseteq \mathrm{Sz}(q)$ and if X does not generate a proper subgroup, *i.e.* if X is not contained in a maximal subgroup. To determine if $g \in X$ is in $\mathrm{Sz}(q)$, first determine if $\det(g) = 1$, then determine if g preserves the symplectic form of $\mathrm{Sp}(4, q)$ and finally determine if g is a fixed point of the automorphism Ψ of $\mathrm{Sp}(4, q)$, mentioned in Section 2.

The recognition algorithm relies on the following result.

Lemma 6.1. *Let $H = \langle X \rangle \leqslant \mathrm{Sz}(q) = G$, where $X = \{x_1, \ldots, x_n\}$ and let $C = \{[x_i, x_j] \mid 1 \leqslant i < j \leqslant n\}$ and M be the natural module of H. Then $H = G$ if and only if the following hold:*

1. *M is an absolutely irreducible H-module.*
2. *H is not conjugate in $\mathrm{GL}(4, q)$ to a subgroup of $\mathrm{GL}(4, r)$, where q is a proper power of r.*
3. *$C \neq \{1\}$ and for every $c \in C \setminus \{1\}$ there exists $x \in X$ such that $[c, c^x] \neq 1$.*

Proof. By Theorem 2.2, the maximal subgroups of G that do not satisfy the first two conditions are $\mathrm{N}_G(\mathcal{H})$, $\mathcal{B}_1$ and $\mathcal{B}_2$. For each, the derived group is contained in the normalised cyclic group, so all these maximal subgroups are metabelian. If H is contained in one of them and H is not abelian, then $C \neq \{1\}$, but $[c, c^x] = 1$ for every $c \in C$ and $x \in X$ since the second derived group of H is trivial. Hence the last condition is not satisfied.

Conversely, assume that $H = G$. Then clearly, the first two conditions are satisfied, and $C \neq \{1\}$. Assume that the last condition is false, so for some $c \in C \setminus \{1\}$

we have that $[c, c^x] = 1$ for every $x \in X$. This implies that $c^x \in \mathrm{C}_G(c) \cap \mathrm{C}_G(c)^{x^{-1}}$, and it follows from Theorem 2.1 that $\mathrm{C}_G(c) = \mathrm{C}_G(c)^{x^{-1}}$. Thus $\mathrm{C}_G(c) = \mathrm{C}_G(c)^g$ for all $g \in G$, so $\mathrm{C}_G(c) \triangleleft G$, but G is simple and we have a contradiction. □

Theorem 6.2. *There exists a Las Vegas algorithm that, given $X \subseteq \mathrm{GL}(4, q)$, decides whether or not $\langle X \rangle = \mathrm{Sz}(q)$. Its time complexity is $\mathrm{O}(|X|^2)$ field operations.*

Proof. The algorithm proceeds as follows.

(1) Determine if every $x \in X$ is in $\mathrm{Sz}(q)$, and return `false` if not.
(2) Determine if $\langle X \rangle$ is absolutely irreducible and if it is not conjugate in $\mathrm{GL}(4, q)$ to a subgroup of $\mathrm{GL}(4, r)$, where q is a proper power of r. Return `false` if any of these turn out to be false.
(3) Using the notation of Lemma 6.1, try to find $c \in C$ such that $c \neq 1$. Return `false` if it cannot be found.
(4) If such c can be found, and if $[c, c^x] \neq 1$ for some $x \in X$, then return `true`, else return `false`.

From the discussion at the beginning of this section, the first step is easily done using $\mathrm{O}(|X|)$ field operations. The MeatAxe (see [13] and [15]) can be used to determine if the natural module is absolutely irreducible; the algorithm of [11] can be used to determine if $\langle X \rangle$ is conjugate in $\mathrm{GL}(4, q)$ to a subgroup of $\mathrm{GL}(4, r)$, where q is a proper power of r. Both these algorithms have time complexity $\mathrm{O}(|X|)$ field operations.

The rest of the algorithm is a straightforward application of the last condition in Lemma 6.1, except that it is sufficient to use the condition for one nontrivial commutator c. By Lemma 6.1, if $[c, c^x] \neq 1$ then $\langle X \rangle = \mathrm{Sz}(q)$; but if $[c, c^x] = 1$, then $C_{\langle X \rangle}(c) \triangleleft \langle X \rangle$ and we cannot have $\mathrm{Sz}(q)$.

It follows immediately that the time complexity of the algorithm is $\mathrm{O}(|X|^2)$ field operations. Since the MeatAxe is Las Vegas, this algorithm is also Las Vegas. □

7. The conjugation problem

Given a conjugate G of $\mathrm{Sz}(q)$ we describe an algorithm to construct an isomorphism from G to $\mathrm{Sz}(q)$ by finding a conjugating element. As one component, we need another recognition algorithm for G, since the one described in Section 6 only works for the standard copy of $\mathrm{Sz}(q)$. In [4], a general recognition algorithm is described which could be used, but we prefer the very fast algorithm described below, which works for this special case.

7.1. **Recognition.** We want to determine if a given group $G = \langle X \rangle \leqslant \mathrm{GL}(4, q)$ is a conjugate of $\mathrm{Sz}(q)$, without finding a conjugating element. We consider carefully the subgroups of $\mathrm{Sp}(4, q)$ and rule out all except those isomorphic to $\mathrm{Sz}(q)$. This relies on the fact that, up to Galois automorphisms, $\mathrm{Sz}(q)$ has only one equivalence class of faithful representations in $\mathrm{GL}(4, q)$ (see [21]), so if we can show that $G \cong \mathrm{Sz}(q)$ then G is a conjugate of $\mathrm{Sz}(q)$.

Theorem 7.1. *There exists a Las Vegas algorithm that, given $X \subseteq \mathrm{GL}(4, q)$, decides whether or not $\langle X \rangle^h = \mathrm{Sz}(q)$ for some $h \in \mathrm{GL}(4, q)$. The algorithm has time complexity $\mathrm{O}(|X|^2)$ field operations.*

Proof. Let $G = \langle X \rangle$. The algorithm proceeds as follows.

(1) Determine if G is absolutely irreducible, using the MeatAxe, and return `false` if not.
(2) Determine if G preserves a non-zero symplectic form M. If so we conclude that G is a subgroup of a conjugate of $\mathrm{Sp}(4, q)$, and if not then return `false`. This is essentially isomorphism testing of modules, which is described in

[13]. Since G is absolutely irreducible, the form is unique up to a scalar multiple.

(3) Conjugate G so that it preserves the form J. This amounts to finding a symplectic basis, *i.e.* finding an invertible matrix X such that $XJX^T = M$, which is easily done. Then G^X preserves the form J and thus $G^X \leqslant \mathrm{Sp}(4, q)$ so that we can apply Ψ.

(4) Determine if $V \cong V^{\Psi}$, where V is the natural module for G and Ψ is the automorphism from Lemma 2.3. If so we conclude that G is a subgroup of some conjugate of $\mathrm{Sz}(q)$, and if not then return `false`.

(5) Determine if G is a proper subgroup of $\mathrm{Sz}(q)$, *i.e.* if it is contained in a maximal subgroup. This can be done using Lemma 6.1. If so, then return `false`, else return `true`.

The algorithms for finding a preserved form and for module isomorphism testing are Las Vegas, with the same time complexity as the MeatAxe (see [13] and [15]), which is $\mathrm{O}(|X|)$ field operations since G has constant degree. Hence we obtain a Las Vegas algorithm, with the same time complexity as the algorithm from Theorem 6.2. □

7.2. **Finding a conjugating element.** Now we assume that we are given $G \leqslant \mathrm{GL}(4, q)$ such that $G^h = \mathrm{Sz}(q)$ for some $h \in \mathrm{GL}(4, q)$, and we turn to the problem of finding some $g \in \mathrm{GL}(4, q)$ such that $G^g = \mathrm{Sz}(q)$, thus obtaining an isomorphism from any conjugate of $\mathrm{Sz}(q)$ to the standard copy.

Lemma 7.2. *Given a random element oracle for subgroups of* $\mathrm{GL}(4, q)$*, there exists a Las Vegas algorithm that, given* $X \subseteq \mathrm{GL}(4, q)$ *such that* $\langle X \rangle^h = \mathrm{Sz}(q)$ *for some* $h \in \mathrm{GL}(4, q)$*, finds a point* $P \in \mathcal{O}^{h^{-1}} = \left\{ Qh^{-1} \mid Q \in \mathcal{O} \right\}$*. The algorithm has time complexity* $\mathrm{O}(\log q)$ *field operations.*

Proof. Clearly $\mathcal{O}^{h^{-1}}$ is the set on which $\langle X \rangle$ acts doubly transitively. For a matrix $M'(\lambda) \in \mathrm{Sz}(q)$ we see that the eigenspaces corresponding to the eigenvalues $\lambda^{\pm(t+1)}$ will be in $\mathcal{O}$. Moreover, every element of order dividing $q - 1$ in every conjugate G of $\mathrm{Sz}(q)$ will have eigenvalues of the form $\mu^{\pm(t+1)}$, $\mu^{\pm 1}$ for some $\mu \in \mathbb{F}_q^{\times}$, and the eigenspaces corresponding to $\mu^{\pm(t+1)}$ will lie in the set on which G acts doubly transitively.

Hence to find a point $P \in \mathcal{O}^{h^{-1}}$ it suffices to find a random $g \in \langle X \rangle$ of order dividing $q - 1$, which is easy by Lemma 2.5, and then find the eigenspaces of g.

Clearly this is a Las Vegas algorithm that uses $\mathrm{O}(\log q)$ field operations. □

Lemma 7.3. *There exists a Las Vegas algorithm that, given* $X \subseteq \mathrm{GL}(4, q)$ *such that* $\langle X \rangle^d = \mathrm{Sz}(q)$ *where* $d = \mathrm{diag}(d_1, d_2, d_3, d_4) \in \mathrm{GL}(4, q)$*, finds a diagonal matrix* $e \in \mathrm{GL}(4, q)$ *such that* $\langle X \rangle^e = \mathrm{Sz}(q)$*, using* $\mathrm{O}(|X| + \log q)$ *field operations.*

Proof. Let $G = \langle X \rangle$. Since $G^d = \mathrm{Sz}(q)$, G must preserve the symplectic form

$$K = dJd = \begin{bmatrix} 0 & 0 & 0 & d_1 d_4 \\ 0 & 0 & d_2 d_3 & 0 \\ 0 & d_2 d_3 & 0 & 0 \\ d_1 d_4 & 0 & 0 & 0 \end{bmatrix} \tag{7.1}$$

where J is given by (2.12). Using [13], we can find this form, which is determined up to a scalar multiple. Hence the diagonal matrix $e = \mathrm{diag}(e_1, e_2, e_3, e_4)$ that we want to find is also determined up to a scalar multiple (and up to multiplication by a diagonal matrix in $\mathrm{Sz}(q)$).

Since e must take J to K, we must have $K_{1,4} = d_1 d_4 = e_1 e_4$ and $K_{2,4} = d_2 d_3 = e_2 e_3$. The matrix e is determined up to a scalar multiple, so we can choose $e_4 = 1$ and $e_1 = K_{1,4}$. Hence it only remains to determine e_2 and e_3.

To conjugate G into $\mathrm{Sz}(q)$ we must have $Pe \in \mathcal{O}$ for every point $P \in \mathcal{O}^{d^{-1}}$, which is the set on which G acts doubly transitively. By Lemma 7.2, we can find $P = (p_1 : p_2 : p_3 : 1) \in \mathcal{O}^{d^{-1}}$, and the condition $Pe = (p_1 K_{1,4} : p_2 e_2 : p_3 e_3 : 1) \in \mathcal{O}$ is given by (2.11) and amounts to

$$p_2 p_3 K_{2,3} + (p_2 e_2)^t + (p_3 e_3)^{t+2} - p_1 K_{1,4} = 0 \tag{7.2}$$

which is a polynomial equation in the two variables e_2 and e_3.

Notice that we can consider e_2^t to be the variable, instead of e_2, since if $x = e_2^t$, then $e_2 = \sqrt{x^t}$. Similarly, we can let e_3^{t+2} be the variable instead of e_3, since if $y = e_3^{t+2}$ then $e_3 = y^{1-t/2}$. Thus instead of (7.2) we obtain a linear equation

$$p_2^t x + p_3^{t+2} y = p_1 K_{1,4} - p_2 p_3 K_{2,3} \tag{7.3}$$

in the variables x, y. Thus the complete algorithm for finding e proceeds as follows.

(1) Find the form K that is preserved by G, using [13].
(2) Find $P, Q \in \mathcal{O}^{d^{-1}}$ using Lemma 7.2.
(3) Let $P = (p_1 : p_2 : p_3 : p_4)$ and $Q = (q_1 : q_2 : q_3 : q_4)$. Determine if the following linear system in the variables x and y is singular, and if so return with failure.

$$\begin{aligned} p_2^t x + p_3^{t+2} y &= p_1 K_{1,4} - p_2 p_3 K_{2,3} \\ q_2^t x + q_3^{t+2} y &= q_1 K_{1,4} - q_2 q_3 K_{2,3} \end{aligned} \tag{7.4}$$

(4) Let (α, β) be a solution to the linear system. The diagonal matrix $e = \operatorname{diag}(K_{1,4}, \sqrt{\alpha^t}, \beta^{1-t/2}, 1)$ now satisfies that $G^e = \mathrm{Sz}(q)$.

By Lemma 7.2 and [13], this is a Las Vegas algorithm that uses $\mathrm{O}(|X| + \log q)$ field operations. □

Lemma 7.4. *There exists a Las Vegas algorithm that, given subsets X, Y_P and Y_Q of $\mathrm{GL}(4,q)$ such that $O_2(G_P) < \langle Y_P \rangle \leqslant G_P$ and $O_2(G_Q) < \langle Y_Q \rangle \leqslant G_Q$, respectively, where $\langle X \rangle = G$, $G^h = \mathrm{Sz}(q)$ for some $h \in \mathrm{GL}(4,q)$ and $P, Q \in \mathcal{O}^{h^{-1}}$, finds $k \in \mathrm{GL}(4,q)$ such that $(G^k)^d = \mathrm{Sz}(q)$ for some diagonal matrix $d \in \mathrm{GL}(4,q)$. The algorithm has time complexity $\mathrm{O}(|X|)$ field operations.*

Proof. Notice that the natural module $V = \mathbb{F}_q^4$ of $\mathcal{FH}$ is uniserial with four non-zero submodules, namely $V_i = \{(v_1, v_2, v_3, v_4) \in \mathbb{F}_q^4 \mid v_j = 0, j > i\}$ for $i = 1, \ldots, 4$. Hence the same is true for $\langle Y_P \rangle$ and $\langle Y_Q \rangle$ (but the submodules will be different) since they lie in conjugates of $\mathcal{FH}$.

Now the algorithm proceeds as follows.

(1) Let $V = \mathbb{F}_q^4$ be the natural module for $\langle Y_P \rangle$ and $\langle Y_Q \rangle$. Find composition series $V = V_4^P \supset V_3^P \supset V_2^P \supset V_1^P$ and $V = V_4^Q \supset V_3^Q \supset V_2^Q \supset V_1^Q$ using the MeatAxe.
(2) Let $U_1 = V_1^P$, $U_2 = V_3^P \cap V_2^Q$, $U_3 = V_2^P \cap V_3^Q$ and $U_4 = V_1^Q$. For each $i = 1, \ldots, 4$, choose $u_i \in U_i$.
(3) Now let k be the matrix such that k^{-1} has u_i as row i, for $i = 1, \ldots, 4$.

We now motivate the second step of the algorithm. Let $(M)_i$ denote the i-th row of a matrix M, and let V_i^P and V_i^Q be as in the algorithm.

We may assume that $Y_P = \{x, y\}$, $Y_Q = \{u, v\}$ where $|x| = |u| = 4$ and both $|y|$ and $|v|$ divide $q - 1$ (and y and v are nontrivial).

There exists $g' \in \mathrm{Sz}(q)$ such that $Phg' = P_\infty$ and $Qhg' = P_0$, since $\mathrm{Sz}(q)$ acts doubly transitively on $\mathcal{O}$. If we let $z = hg'$, then $\langle Y_P \rangle^z$ and $\langle Y_Q \rangle^z$ consist of lower and upper triangular matrices, respectively. Hence there exist $a_1, b_1 \in \mathbb{F}_q$ such that $x = S(a_1, b_1)^{z^{-1}}$, and then $V_1^P = \langle (x)_1 \rangle = \langle (S(a_1, b_1))_1 z^{-1} \rangle = V_1$. But $(S(a_1, b_1))_1 z^{-1} = (z^{-1})_1$ so by choosing some non-zero vector in V_1^P we obtain a scalar multiple of the first row of z^{-1}. Similarly, there exist $a_2, b_2 \in \mathbb{F}_q$ such that $u = (S(a_2, b_2)^T)^{z^{-1}}$, and $V_1^Q = \langle (u)_4 \rangle = \langle (S(a_2, b_2)^T)_4 z^{-1} \rangle$, where $S(a_2, b_2)^T$ is the transpose of $S(a_2, b_2)$. But $(S(a_2, b_2)^T)_4 z^{-1} = (z^{-1})_4$ so by choosing some non-zero vector in V_1^Q we obtain a scalar multiple of the fourth row of z^{-1}.

Note that $\dim V_3^P \cap V_2^Q = 1$ and $\dim V_2^P \cap V_3^Q = 1$, and by choosing non-zero vectors from these we obtain scalar multiples of the second and third rows of z^{-1}, respectively.

Thus the matrix k found in the algorithm satisfies that $z = kd$ for some diagonal matrix $d \in \mathrm{GL}(4, q)$. Since $\mathrm{Sz}(q) = G^h = G^z = (G^k)^d$, the algorithm returns a correct result, and it is Las Vegas because the MeatAxe is Las Vegas (see [13] and [15]). Clearly the time complexity is the same as the MeatAxe, so the algorithm uses $\mathrm{O}(|X|)$ field operations. $\square$

Theorem 7.5. *Assuming Conjecture 4.2 and given a random element oracle for subgroups of* $\mathrm{GL}(4, q)$*, there exists a Las Vegas algorithm that, given* $X \subseteq \mathrm{GL}(4, q)$ *such that* $\langle X \rangle^h = \mathrm{Sz}(q)$ *for some* $h \in \mathrm{GL}(4, q)$*, finds* $g \in \mathrm{GL}(4, q)$ *such that* $\langle X \rangle^g = \mathrm{Sz}(q)$*. The algorithm has time complexity* $\mathrm{O}(\log(q) \log\log(q) + |X|)$ *field operations.*

Proof. Let $G = \langle X \rangle$. First note that g is determined up to multiplication by an element of $\mathrm{Sz}(q)$, so we will find g such that $hg' = g$ where $g' \in \mathrm{Sz}(q)$.

The algorithm described in Corollary 4.8 works equally well for a conjugate of $\mathrm{Sz}(q)$, so we can find generators for a stabiliser of a point in G, using the algorithm described in Theorem 5.2. In this case we do not need the elements as SLPs, so a discrete log oracle is not necessary.

(1) Find points $P, Q \in \mathcal{O}^{h^{-1}}$ using Lemma 7.2. Return with failure if $P = Q$.
(2) Find generating sets Y_P and Y_Q such that $O_2(G_P) < \langle Y_P \rangle \leqslant G_P$ and $O_2(G_Q) < \langle Y_Q \rangle \leqslant G_Q$ using the first three steps of the algorithm from the proof of Theorem 5.2.
(3) Find $k \in \mathrm{GL}(4, q)$ such that $(G^k)^d = \mathrm{Sz}(q)$ for some diagonal matrix $d \in \mathrm{GL}(4, q)$, using Lemma 7.4.
(4) Find a diagonal matrix e using Lemma 7.3.
(5) Now $g = ke$ satisfies that $G^g = \mathrm{Sz}(q)$.

Be Lemma 7.2, 7.4 and 7.3, and the proof of Theorem 5.2, this is a Las Vegas algorithm with time complexity as stated. $\square$

8. Implementation and performance

An implementation of the algorithms described here is available in Magma. The implementation uses the existing Magma implementations of the algorithms described in [6], [7], [11], [13] and [24, Corollary 14.16].

A benchmark of the recognition algorithm described in Section 7.1, for various field sizes $q = 2^{2m+1}$, is given in Figure 8.1. For each field size, 200 random conjugates of $\mathrm{Sz}(q)$ were recognised and the average running time for each call is displayed.

A benchmark of the conjugation algorithm described in Section 7.2, for various field sizes $q = 2^{2m+1}$, is given in Figure 8.2. For each field size, 100 random conjugates of $\mathrm{Sz}(q)$ were considered and a conjugating element found. The average running time for each call is displayed.

The constructive membership and conjugation algorithms both need to compute generating sets of stabilisers, so they depend on Algorithm 1. Therefore our implementation depends on the Magma implementation of discrete log. Since we are in characteristic 2, there is a specialised algorithm for discrete log, *Coppersmith's algorithm* (see [9]), which is implemented in Magma.

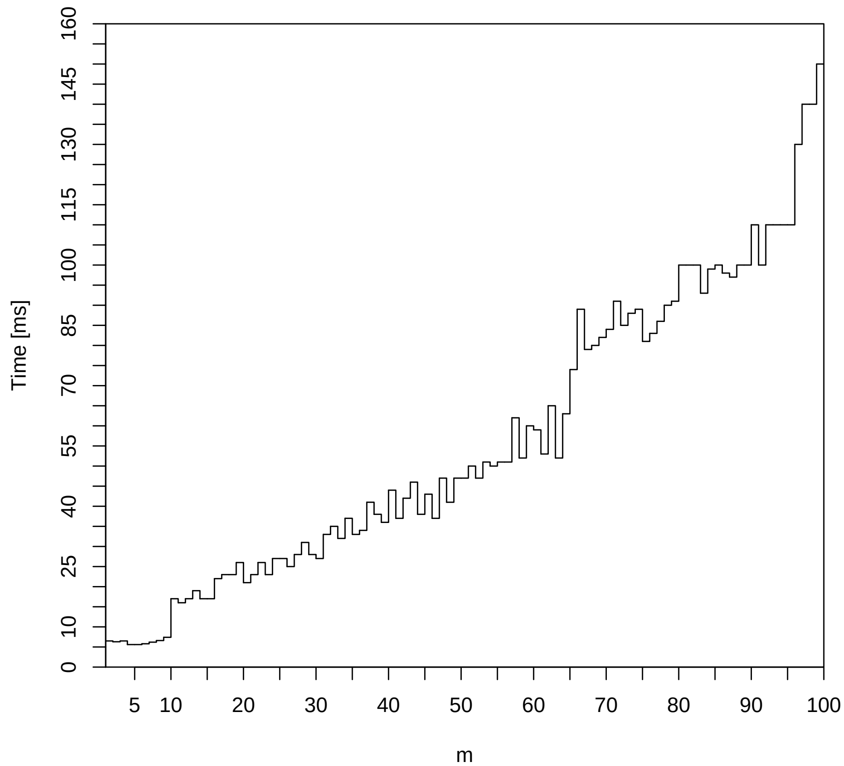

FIGURE 8.1. Benchmark of recognition

We have benchmarked the computation of generating sets for stabilisers, for various field sizes, as shown in Figure 8.3. For each field size, $q = 2^{2m+1}$, generating sets for the stabilisers of 100 random points were computed, and the average running time for each call is listed. The amount of this time that was spent in discrete logarithm computations is also indicated.

We used the software package R (see [19]), to produce Figures 8.1, 8.2 and 8.3.

All benchmarks were carried out using MAGMA V2.12-9, on a PC with an Intel Xeon CPU running at 2.8 GHz and with 1 GB of RAM. For the conjugation problem, the highest value of m was 55, since higher field sizes required too much memory. For the recognition and stabiliser computation, there was never any shortage of memory, and the benchmark indicated that much larger fields should also be feasible. The expectation was that the conjugation problem and the stabiliser computation would be much more time consuming than the recognition, and in order to shorten the total time, 100 rather than 200 computations were performed for each field size. The benchmark confirmed this expectation.

Moreover, the benchmark was also used as a way to check Conjecture 4.2. Each stabiliser computation involves at least 2 calls to Algorithm 1, so at least 14000 checks of the conjecture was made during the benchmark. The fact that it never failed provides strong evidence to support the conjecture.

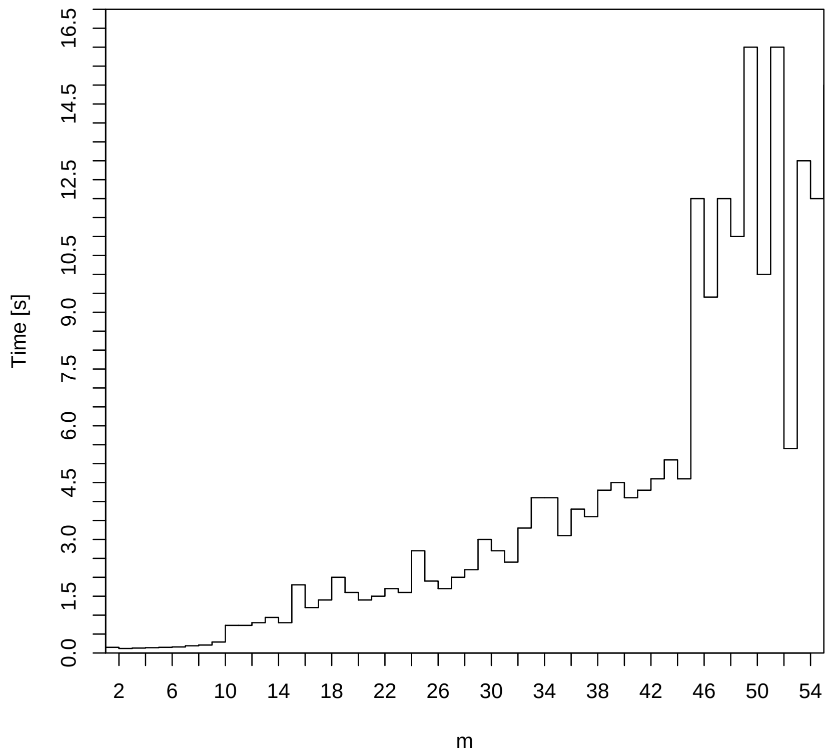

FIGURE 8.2. Benchmark of conjugation

REFERENCES

1. M. Aschbacher, *On the maximal subgroups of the finite classical groups*, Invent. Math. **76** (1984), 469–514.
2. H. Bäärnhielm, *Tensor decomposition of the Suzuki groups*, (2005), submitted.
3. L. Babai, *Local expansion of vertex-transitive graphs and random generation in groups*, Proc. 23rd ACM Symp. Theory of Computing (Los Angeles), Association for Computing Machinery, 1991, pp. 164–174.
4. L. Babai, W. M. Kantor, P. P. Pálfy, 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.
5. W. Bosma, J. Cannon, and C. Playoust, *The Magma algebra system I: The user language*, J. Symbolic Comput. **24** (1997), 235–265.
6. F. Celler and C. R. Leedham-Green, *Calculating the order of an invertible matrix*, Groups and Computation II (Larry Finkelstein and William M. Kantor, eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 28, American Mathematical Society, 1997, pp. 55–60.
7. 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. Algebra (1995), no. 23, 4931–4948.
8. M. D. E. Conder, C. R. Leedham-Green, and E. A. O'Brien, *Constructive recognition of* $PSL(2, q)$, Trans. Amer. Math. Soc. **358** (2006), 1203–1221.
9. D. Coppersmith, *Fast evaluation of logarithms in fields of characteristic two*, IEEE Trans. Inform. Theory **IT-30** (1984), no. 4, 587–594.
10. C. W. Curtis and I. Reiner, *Representation theory of finite groups and associative algebras*, John Wiley & Sons, 1962.
11. S. P. Glasby, C. R. Leedham-Green, and E. A. O'Brien, *Writing projective representations over subfields*, J. Algebra **295** (2006), 51–61.

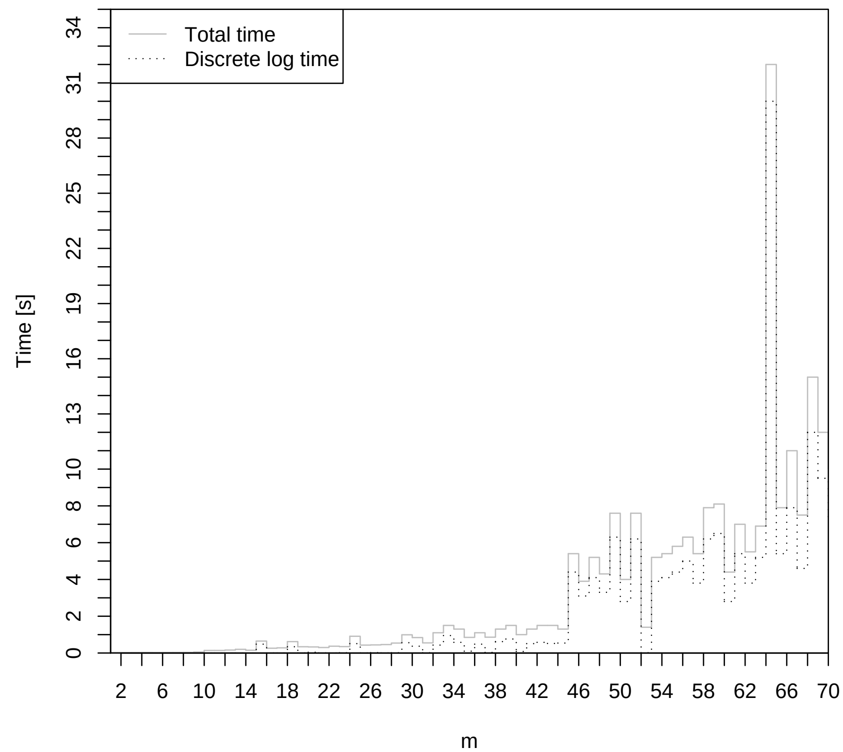

FIGURE 8.3. Benchmark of stabiliser computation

12. D. F. Holt, B. Eick, and E. A. O'Brien, *Handbook of Computational Group Theory*, Chapman & Hall/CRC, January 2005.
13. D. F. Holt and S. Rees, *Testing modules for irreducibility*, J. Austral. Math. Soc. Series A **57** (1994), 1–16.
14. B. Huppert and N. Blackburn, *Finite groups III*, Grundlehren Math. Wiss., vol. 243, Springer-Verlag, Berlin, Heidelberg, New York, 1982.
15. G. Ivanyos and K. Lux, *Treating the exceptional cases of the MeatAxe*, Experiment. Math. **9** (2000), 373–381.
16. C. R. Leedham-Green, *The computational matrix group project*, Groups and Computation III, Ohio State Univ. Math. Res. Inst. Publ., vol. 8, de Gruyter, 2001, pp. 113–121.
17. D. S. Mitrinovic, J. Sándor, and B. Crstici, *Handbook of number theory, mathematics and its applications*, vol. 351, Kluwer Academic Publishers, 1996.
18. T. Ono, *An identification of Suzuki groups with groups of generalized Lie type*, Ann. of Math. **75** (1962), no. 2, 251–259.
19. R Development Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2005, 3-900051-07-0.
20. Á. Seress, *Permutation group algorithms*, Cambridge Tracts in Mathematics, vol. 152, Cambridge University Press, 2003.
21. R. Steinberg, *Representations of algebraic groups*, Nagoya Math. J. **22** (1963), 33–56.
22. ______, *On theorems of Lie-Kolchin, Borel and Lang*, Constributions to algebra (collection of papers dedicated to Ellis Kolchin), Academic Press, New York, 1977, pp. 349–354.
23. M. Suzuki, *On a class of doubly transitive groups*, Ann. of Math. **75** (1962), no. 1, 105–145.
24. J. von zur Gathen and J. Gerhard, *Modern computer algebra*, 2nd ed., Cambridge University Press, Cambridge, 2003.
25. R. A. Wilson, *Finite simple groups*, preprint, 2005.

School of Mathematical Sciences, Queen Mary, University of London, Mile End Road, London E1 4NS, United Kingdom

URL: http://www.maths.qmul.ac.uk/~hb/

E-mail address: h.baarnhielm@qmul.ac.uk