Towards effective algorithms for linear groups

*E.A. O'Brien**

Abstract. One of the major research directions in computational group theory over the past 15 years has been the development of effective algorithms for the investigation of subgroups of $\mathrm{GL}(d, F)$ where F is a finite field. We survey this work.

2000 Mathematics Subject Classification: 20C20, 20C40

1. Introduction

Research activity in computational group theory has concentrated on four primary areas: permutation groups, finitely-presented groups, polycyclic groups, and representation theory.

It is now possible in practice to study the structure of permutation groups having degrees up to about ten million; see Seress [77] for further detail. We can readily compute useful descriptions (for certain quotients) of "large" finitely-presented groups; see Sims [80] for further detail. Effective algorithms for the study of (finite and infinite) polycyclic groups have been developed; see [48, Chapter 8] for further detail.

While the study of a group via its modular representations is a fundamental area of mathematical research, limited tools exist for such structural investigation. Consider $G = \langle X \rangle \leq \mathrm{GL}(d, F)$ where $F = \mathrm{GF}(q)$. Natural questions arise. What is the order of G? What are its composition factors? What are its Sylow subgroups? While similar questions about a subgroup of S_n, the symmetric group of degree n, can be answered theoretically and practically using highly effective polynomial-time algorithms, existing machinery for linear groups is much weaker. For example, it is difficult to determine (using existing standard functions) the order of a random subgroup of $\mathrm{GL}(6, 5^2)$ using either of the major computational group theory systems, GAP [38] and MAGMA [14].

*I am most grateful to Bill Kantor for his personal and professional support over many years. I thank the organisers for the invitation and financial support to participate in the meeting. This work was partially supported by the Marsden Fund of New Zealand via grant UOA124. I thank Peter Brooksbank, Derek Holt, Alice Niemeyer, Cheryl Praeger, Ákos Seress, and the referee for their careful reading, comments, and corrections to the paper.

A major topic of research over the past 15 years has been the development of effective well-understood algorithms for the study of such groups. An associated goal is to realise the performance of these algorithms in practice.

One measure of performance is that an algorithm is *polynomial in the size of the input*; for $G = \langle X \rangle \leq \mathrm{GL}(d, q)$, the size of the input is $O(|X| d^2 \log q)$. For a discussion of complexity-related issues, see Seress [77].

1.1. Basic tasks.

Already the most basic computations are expensive for linear groups. Consider multiplying two $d \times d$ matrices. Its complexity is $O(d^\omega)$ field operations, where $\omega = 3$ if we employ the traditional algorithm. Strassen's divide-and-conquer algorithm [82] reduces ω to $\log_2 7$. However, its MAGMA implementation demonstrates better performance over the traditional method only for matrices defined over finite fields having degrees in the hundreds. Further, there are overheads: the additional complexity of the implementation and memory used. While Coppersmith & Winograd [37] demonstrate that ω can be smaller than 2.376, this seems of limited practical significance.

Observe that we can compute large powers m of a matrix g in at most $2 \lfloor \log_2 m \rfloor$ multiplications by the standard recursive algorithm: $g^m = g^{m-1} g$ if m is odd and $g^m = g^{(m/2)2}$ if m is even.

The standard algorithm to compute the characteristic polynomial of a matrix has complexity $O(d^3)$ [48, p. 227]. Storjohann [81] presents a deterministic algorithm having similar complexity to determine its minimal polynomial; a simpler randomised alternative having worst-case complexity $O(d^4)$ is described by Celler & Leedham-Green [28].

1.2. Randomised and black-box algorithms.

Most of the algorithms for linear groups are *randomised*: they rely on random selections, and the analysis of their performance assumes that we can select uniformly distributed random elements.

A *Monte Carlo* algorithm is a randomised algorithm which may return an incorrect answer to a decision question, and the probability of this event is less than some specified value. A *Las Vegas* algorithm is one which never returns an incorrect answer, but may report failure with probability less than some specified value. If one of the answers given by a Monte Carlo algorithm is always correct, then it is *one-sided*. For a discussion of (concepts related to) these types of algorithms, we refer the reader to Babai [5].

Babai [4] presented a Monte Carlo algorithm to construct in polynomial time nearly uniformly distributed random elements of a finite group. No effective implementation of this algorithm is available. Instead, both GAP and MAGMA use the *product replacement algorithm* of Celler *et al.* [27]. That this is also polynomial time was established by Pak [73]. For a discussion of both algorithms, we refer

the reader to [77, pp. 26-30]. Leedham-Green & O'Brien [60] present a variation of the latter algorithm to construct random elements of a normal subgroup.

The concept of a *black-box group* was introduced by Babai & Szemerédi [10]. In this model, group elements are represented by bit-strings of uniform length; the only group operations permissible are multiplication, inversion, and checking for equality with the identity element.

Seress [77, p. 17] defines a *black-box algorithm* as one which does not use specific features of the group representation, nor particulars of how group operations are performed; it can only use the operations listed above. Some of the algorithms surveyed here were first developed in the black-box context, usually under the assumption that *oracles* to perform certain tasks are available.

One such is an *order oracle* to compute the order of an arbitrary element of a group. In Section 2 we describe such an oracle for a linear group.

Another is a *discrete log oracle* which will provide, for a given non-zero element μ of $\mathrm{GF}(q)$ and a fixed primitive element a of $\mathrm{GF}(q)$, the unique integer k in the range $1 \leq k < q$ for which $\mu = a^k$. For a description of discrete log algorithms, see [78, Chapter 4].

Seress [77, Chapter 2] provides an excellent account of black-box algorithms: these include Monte Carlo algorithms to compute the normal closure of a subgroup and to construct the derived group of a black-box group.

One may intuitively think of a *straight-line program* for $g \in G = \langle X \rangle$ as an efficiently stored group word on X that evaluates to g. While the length of a word in a given generating set constructed in n multiplications and inversions can increase exponentially with n, the length of the corresponding straight-line program is *linear* in n. Babai & Szemerédi [10] prove that every element of a finite group G has a straight-line program of length at most $O(\log^2 |G|)$. In practice, both Magma and GAP exploit straight-line programs. We do not explicitly consider the concept further here, but refer the reader to Seress [77] for a discussion of its theoretical and practical significance, particularly in evaluating homomorphisms and relations.

1.3. The approaches.

The *black-box group approach*, initiated by Babai & Beals [7], seeks to determine the abstract group-theoretic structure of G. The associated algorithms are black-box, usually Monte Carlo.

Every finite group G has a series of characteristic subgroups

$$1 \leq O_\infty(G) \leq \mathrm{Soc}^*(G) \leq \mathrm{Pker}(G) \leq G,$$

where $O_\infty(G)$ is the largest soluble normal subgroup of G. Here $\mathrm{Soc}^*(G)/O_\infty(G)$ is the socle of the factor group $G/O_\infty(G)$, and so $\mathrm{Soc}^*(G)/O_\infty(G)$ is isomorphic to a direct product $T_1 \times \cdots \times T_k$ of nonabelian simple groups that are permuted by conjugation in G; further $\mathrm{Pker}(G)$ is the kernel of this permutation action. For a more detailed account of this structure, see, for example, [48, pp. 31–32].

Given $G = \langle X \rangle \leq \mathrm{GL}(d, q)$, Babai & Beals [7] present a Monte Carlo algorithm to construct subgroups $H_1, \ldots, H_k$ such that $H_i/O_\infty(H_i) \cong T_i$, or H_i acts on a permutation domain of size polynomial in d. If we can construct the H_i, then we can construct $G/\mathrm{Pker}(G) \leq S_k$, which can be studied readily using permutation group methods. The remaining theoretical difficulty is the construction of (generators for) the soluble radical of G in Monte Carlo polynomial time.

By contrast, the *geometric approach* seeks to investigate whether a linear group G satisfies certain natural and inherent properties in its action on its underlying vector space. If so, it determines an *Aschbacher category* of G, identifies an $N \lhd G$ naturally associated with this category, and recursively studies G/N and N. Our primary focus in this survey is the geometric approach.

Luks [65] proved that we can decide solubility for linear groups in polynomial time, and presented *deterministic* algorithms to answer a variety of questions for soluble linear groups. These algorithms are polynomial, not in the size of the input group G, but in the *largest prime divisor* of $|G|$ other than the characteristic. This work has been developed and extended by Miyazaki [66]. While Cooperman and O'Brien developed a prototype implementation of Luks' algorithm in 2000, its full potential has not yet been practically realised.

1.4. The major tasks.

In designing algorithms for the structural investigation of a *simple* group $G = \langle X \rangle$, we identify three natural and significant tasks.

- Determine the name of G.
- Construct an isomorphism between G and a "standard" copy of G.
- Given $g \in G$, write g as a word in X: with considerable abuse of notation, we say that this task is the *word-problem* for G.

Two major types of algorithms have been developed to solve these tasks. A *non-constructive recognition algorithm* names G. (More precisely, it may simply establish that G *contains* a particular named group as a composition factor.) Clearly such an identification is useful. If, for example, we identify G as a member of a particular family of finite simple groups, then we may apply algorithms to G which are specially designed for this family.

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

For example, if G is an alternating group of degree n, then a constructive recognition algorithm sets up an isomorphism between G and the standard copy H on n points generated by a 3-cycle and an $(n-1)$- or n-cycle.

Two algorithms which solve the word-problem for a given group, but do not (readily) fit the constructive recognition model, are outlined in Sections 7.4 and 7.5.

As part of their ongoing work on groups of Lie type, Cohen, Murray & Taylor [31] developed the *generalised row and column reduction algorithm*: for certain matrix representations, this algorithm writes an element of a group of Lie type as a word in its Steinberg generators. This is *one* component of a solution to the word-problem for these groups. (Of course, we must first construct the Steinberg generators as words in the defining generators of the input group.)

1.5. An overview.

We aim to provide an introduction to this research topic; both its high level of activity and our current state of knowledge dictate that this is a report of "work in progress". For an excellent survey of related topics, see Kantor & Seress [55].

While it is still too early to predict the final outcome of "matrix group recognition", we believe that a realistic and achievable goal is to provide effective well-understood algorithms to answer many questions for linear groups of "small" degree. The principal outstanding practical obstacle is constructive recognition for classical groups, presented as matrix groups in defining characteristic. Increasingly, the division between the two approaches sketched in Section 1.3 is artificial. While some algorithms are developed in a black-box context, usually under the assumption that oracles to perform certain tasks are available, their implementations accept as input a linear group or a permutation group, where algorithms which are not black-box perform such tasks. Further, Mark Stather and others already exploit ideas from both approaches, and we expect that some mixture will ultimately prove most effective at a practical level.

In Section 2 we describe an order oracle for a linear group. Aschbacher's classification of maximal subgroups of classical groups into nine categories is summarised in Section 3. Section 4 surveys existing algorithms to decide membership of the categories, and in Section 5 we discuss how to exploit the associated geometry. In Section 6 we survey non-constructive algorithms which name the finite simple groups, and in Section 7 survey algorithms which solve the word-problem for these groups. Finally, we consider *short presentations* for simple groups, which may be used to verify that the results of randomised algorithms are correct.

2. Determining orders

A natural question is: *determine the order of* $g \in \mathrm{GL}(d, q)$. The task currently requires factorisation of numbers of the form $q^i - 1$, a problem generally believed not to be solvable in polynomial time. (Since $\mathrm{GL}(d, q)$ has elements of order $O(q^d)$, we cannot simply compute powers of g until we obtain the identity!)

Celler & Leedham-Green [28] present the following algorithm to compute the order of $g \in \mathrm{GL}(d, q)$.

- Compute a "good" multiplicative upper bound E for $|g|$.

- Now factorise $E = \prod_{i=1}^{m} p_i^{\alpha_i}$ where the primes p_i are distinct.

- If $m = 1$, then calculate $g^{p_1^j}$ for $j = 1, 2, \ldots, \alpha_1 - 1$ until the identity is constructed.

- If $m > 1$ then express $E = uv$, where u, v are coprime and have approximately the same number of distinct prime factors. Now g^u has order k dividing v and g^k has order ℓ say dividing u, and the order of g is $k\ell$. Hence the algorithm proceeds by recursion on m.

How do we obtain a good multiplicative upper bound? Given g, determine and factorise its minimal polynomial $f(x) = \prod_{i=1}^{t} f_i(x)^{m_i}$ where $\deg(f_i) = d_i$. Now $\beta = \log_p \max m_i$ and set

$$E = \operatorname{lcm}(q^{d_1} - 1, \ldots, q^{d_t} - 1) \times p^{\beta}.$$

Observe that $|g|$ divides E. Celler & Leedham-Green [28] prove the following:

Theorem 2.1. *If we know a factorisation of E, the cost of the order algorithm is $O(d^3 \log q \log\log q^d)$ field operations.*

If we fail to complete the factorisation of E, then we obtain a *pseudo-order* for g – namely, a multiple of its order by some large prime(s). For most theoretical and practical purposes this suffices.

Implementations of the algorithm in both GAP and Magma use databases of factorisations of numbers of the form $q^i - 1$, prepared as part of the Cunningham Project [18].

A related problem is the following. Let G be a black-box group having an order oracle, and let $N \lhd G$: *determine the order of an element of G/N.*

Leedham-Green & O'Brien [60] present an algorithm for this task. Let $g \in G$ and let m be its order. The basic algorithm iterates the following operation for some preassigned number of times.

- $a :=$ random element of N;

- $m := \gcd(m, |ga|)$;

It then returns m as the estimate of the order of the image of g in G/N.

If the basic algorithm returns $m > 1$, we apply the following refinement. For every prime p dividing m, apply the basic algorithm to $g^{m/p}$. If the algorithm returns 1 or any number prime to p as the order of the image of $g^{m/p}$, then the order of the image of g divides m/p; now repeat this refinement with m replaced by m/p.

Babai & Shalev [9] prove the following:

Lemma 2.2. *Let N be a simple non-abelian normal subgroup of G. The refined algorithm, with high probability, returns the order of g modulo N as 1 if $g \in N$.*

Hence this algorithm can decide *membership* in a normal subgroup (provably so for one which is simple), and thus is important for working with quotients of black-box groups.

A consequence, of practical and theoretical importance, is a one-sided Monte Carlo algorithm to prove that a black-box group G is perfect: we prove that every generator of G is an element of its derived group and so learn that G is perfect. An implementation is available in MAGMA, and is used extensively in our implementation of the identification algorithm of Section 6.2.

3. Geometry following Aschbacher

As mentioned in the introduction, a classification of the maximal subgroups of $\mathrm{GL}(d,q)$ by Aschbacher [3] underpins the "geometric" approach to the study of linear groups. Let Z denote the group of scalar matrices of G. Then G is *almost simple modulo scalars* if there is a non-abelian simple group T such that $T \leq G/Z \leq \mathrm{Aut}(T)$, the automorphism group of T.

We summarise Aschbacher's classification as follows: a linear group preserves some natural linear structure in its action on the underlying space and has a normal subgroup related to this structure, or it is almost simple modulo scalars.

More formally, we paraphrase the theorem as follows.

Theorem 3.1. *Let V be the vector space of row vectors on which $\mathrm{GL}(d,q)$ acts, and let Z be the subgroup of scalar matrices of G. If G is a maximal subgroup of $\mathrm{GL}(d,q)$, then one of the following is true:*

C1. *G acts reducibly.*

C2. *G acts imprimitively: G preserves a decomposition of V as a direct sum $V_1 \oplus V_2 \oplus \cdots \oplus V_r$ of $r > 1$ subspaces of dimension s, which are permuted transitively by G, and so $G \leq \mathrm{GL}(s,q) \wr \mathrm{Sym}(r)$.*

C3. *G acts on V as a group of semilinear automorphisms of a (d/e)-dimensional space over the extension field $\mathrm{GF}(q^e)$, for some $e > 1$ and so G embeds in $\Gamma L(d/e, q^e)$. (This includes the class of "absolutely reducible" linear groups, where G embeds in $\mathrm{GL}(d/e, q^e)$.)*

C4. *G preserves a decomposition of V as a tensor product $U \otimes W$ of spaces of dimensions $d_1, d_2 > 1$ over F. Then G is a subgroup of the central product of $\mathrm{GL}(d_1, q)$ and $\mathrm{GL}(d_2, q)$.*

C5. *G is definable modulo scalars over a subfield: for some proper subfield $\mathrm{GF}(q')$ of $\mathrm{GF}(q)$, $G^g \leq \mathrm{GL}(d, q').Z$, for some $g \in \mathrm{GL}(d,q)$.*

C6. *For some prime r, $d = r^m$ and G/Z is contained in the normaliser of an extraspecial group of order r^{2m+1}, or of a group of order 2^{2m+2} and symplectic-type.*

C7. *G is tensor-induced: it preserves a decomposition of V as $V_1 \otimes V_2 \otimes \cdots \otimes V_m$, where each V_i has dimension $r > 1$ and the set of V_i is permuted by G, and so $G/Z \leq \mathrm{PGL}(r,q) \wr \mathrm{Sym}(m)$.*

C8. *G normalises a classical group in its natural representation.*

C9. *G is almost simple modulo scalars.*

Of course, the nine Aschbacher categories are not mutually exclusive. Further, seven have a normal subgroup associated with a decomposition.

In broad outline, this theorem suggests that a first step in investigating a linear group is to determine (at least one of) its categories in the Aschbacher classification. If a category is recognised, then we can investigate the group structure more completely using algorithms designed for this category. Usually, we have reduced the size and nature of the problem. For example, if $G \leq \mathrm{GL}(d,q)$ acts imprimitively, then we obtain a permutation representation of degree at most d for G; if G preserves a tensor product, we now consider two linear groups of smaller degree. If a proper normal subgroup N exists, we recognise N and G/N recursively, ultimately obtaining a composition series for G. Many questions about the structure of G can be answered by first considering its composition factors.

What of the almost simple groups? Liebeck [62] proved that the maximal non-classical subgroups of $\mathrm{GL}(d,q)$ have order at most q^{3d}, small by comparison with $\mathrm{GL}(d,q)$ which has order $O(q^{d^2})$.

Further, the absolutely irreducible representations of degree at most 250 of all quasisimple finite groups are now explicitly known: see Hiss & Malle [43] and Lübeck [64]. (Recall that G is *quasisimple* if G is perfect and $G/Z(G)$ is simple.) The algorithmic potential of these lists remains to be realised.

4. Membership of an Aschbacher category

We survey work on deciding if $G \leq \mathrm{GL}(d,F)$, where $F = \mathrm{GF}(q)$, acting on the underlying vector space V, is a member of the first seven Aschbacher categories. In Section 6.1 we report on a Monte-Carlo algorithm which decides if G is in C8. We first consider an algorithm which plays an important role in such investigations.

4.1. The Smash algorithm.

In essence, the Smash algorithm presented in [46] is a constructive realisation of Clifford's theorem [30].

Assume that G acts absolutely irreducibly on V. Let $S \subseteq G$ contain at least one non-scalar element. In summary, this algorithm investigates whether G has certain decompositions with respect to the normal closure $\langle S \rangle^G$. The possible decompositions correspond to categories in Aschbacher's theorem.

We now consider these in more detail. Let N be a normal non-scalar subgroup of G. Then, for some $t \geq 1$, V splits as a direct sum $W_1 \oplus W_2 \oplus \cdots \oplus W_t$ of irreducible FN-modules, all of the same dimension. For some $r, s' \geq 1$, with $rs' = t$, the W_is partition into r sets containing s' pairwise isomorphic FN-modules each. If $V_1, V_2, \ldots, V_r$ are each the sum of s' pairwise isomorphic W_is, so that $V = V_1 \oplus V_2 \oplus \cdots \oplus V_r$, then G permutes the V_is transitively. Four situations arise:

- If $r > 1$ then G acts imprimitively on V (type C2).
- If $r = 1$ and $t > 1$ and the W_i are absolutely irreducible as FN-modules, then V can be recognised as a tensor product preserved by G (type C4).
- If $r = 1$ and the W_i are not absolutely irreducible as FN-modules, then G is semilinear (type C3).
- Otherwise, both r and t equal 1 and N acts absolutely irreducibly on V. Now $N/Z(N)$ is a direct product $N_0 \times N_0 \times \cdots \times N_0$ of m copies of a simple group N_0, and N is a central product of m groups N_1, each isomorphic to an extension of $Z(N)$ by N_0. If N_0 is cyclic, then G normalises an extraspecial or symplectic-type group (type C6). Otherwise N_0 is non-abelian simple. If $m = 1$, G is almost simple, and Smash fails to find a decomposition; otherwise $m > 1$ and G is tensor-induced (type C7).

The complexity of the resulting algorithm is at worst $O(d^6)$ [46]. An implementation is distributed with Magma.

4.2. Reducible groups.

The maximal subgroups in this category are the maximal parabolic subgroups.

If the action of G on V is unipotent, then it is easy to diagonalise G and we find a composition series for G by elementary linear algebra. If there is a proper section S of V on which G acts non-trivially, then we write down the action of G on S; the kernel of the resulting homomorphism is the subgroup of G which centralises this section.

The MeatAxe is a one-sided Monte Carlo algorithm to decide whether or not G acts irreducibly on V. The original algorithm, incorporating ideas of Norton and Parker, is described in [74]. It was generalised and analysed by Holt & Rees [45], a task completed by Ivanyos & Lux [52]. In summary, their algorithm is the following. Let M denote the FG-module and let A denote the F-algebra spanned by the generators of G. Select a random element θ of A, determine its characteristic polynomial $c(x)$ of θ, and factorise it. Let $\chi = p(\theta)$ where $p(x)$ is an irreducible factor of $c(x)$. Hence χ has non-trivial nullspace N. If $p(x)$ is a factor of multiplicity one, then N is irreducible as an $F\langle\theta\rangle$-module. Now compute the FG-submodule of M generated by a single non-zero vector in N. If we obtain a proper submodule, we conclude that G acts reducibly on V; otherwise we must repeat the random selection a number of times.

The MeatAxe has complexity $O(d^{3.5} \log q)$ [45], [52]. Implementations are distributed with GAP and Magma.

4.3. Imprimitive groups.

Groups in this category act irreducibly but imprimitively on V; maximal subgroups in this category are stabilisers of direct sum decompositions $V = \oplus_{i=1}^{r} V_i$ where $\dim(V_i) = d/r = s$. (A space V_i is a *block*, the set $\{V_1, \ldots, V_r\}$ is a *block system.*)

If G stabilises such a decomposition, then we obtain a homomorphism $\phi : G \longmapsto \mathrm{Sym}(r)$ and its kernel is a normal subgroup of G.

Holt *et al.* [47] present an algorithm to decide if an absolutely irreducible group G acts imprimitively on its underlying space V.

One of its key components is the MinBlocks algorithm: given a non-trivial subspace of a block of imprimitivity, the algorithm finds the block system with minimal block dimension that contains this subspace.

The Smash algorithm of Section 4.1 applies when G does not act faithfully on the system of blocks. If G has a block system containing r blocks of dimension s, then there is a homomorphism from G to S_r. From a consideration of element orders and characteristic polynomials, we may discover that a particular non-scalar $g \in G$ must lie in the kernel of the homomorphism from G to S_r. If so, we construct its normal closure $N = \langle g \rangle^G$, and then search for a decomposition with respect to N.

If G acts faithfully as a permutation group on the blocks, then we seek to construct the stabiliser of a block.

Suppose that G acts imprimitively on V with blocks of dimension s, and let H be the stabiliser of one such block, W. Our strategy attempts to find H and W, or to establish that the assumption is false. If W exists, then V is isomorphic to the induced module W^G, where W is regarded as an FH-module. Thus, W must be irreducible as an FH-module, since otherwise V would not be irreducible as an FG-module. From [50, Chapter V, Satz 16.6], we have $\mathrm{Hom}_{FG}(W^G, V) \cong \mathrm{Hom}_{FH}(W, V)$. Since we assume that V is an absolutely irreducible FG-module, $\mathrm{Hom}_{FG}(W^G, V)$ has dimension 1 over F. It follows that the only FH-submodule of V that is isomorphic to W is W itself.

This suggests that we try to construct the stabiliser, H, of a fixed but unknown block, W, of dimension s. If we succeed in constructing H, then we can find W by first applying the MeatAxe algorithm to the action of H on V, and then, for each FH-composition factor V_i of dimension s, calculating $\mathrm{Hom}_{FH}(V_i, V)$. If $\mathrm{Hom}_{FH}(V_i, V)$ has dimension one, then W is the unique image in V of every nonzero homomorphism, and we can find the block system by applying MinBlocks to this image.

We may assume that the permutation action of G on the blocks is primitive, and so H must be a maximal subgroup of G of index r. We try to construct H by working up a chain of subgroups, starting with a cyclic subgroup and then adjoining new generators. At some point in our construction, our investigations

may prove that no such H exists, and so we can conclude that G does not preserve a block system with block dimension s.

An implementation of the algorithm is distributed with MAGMA.

4.4. Semilinear groups.

Groups in this category preserve on V the structure of a vector space over an extension field of $\mathrm{GF}(q)$ and maximal subgroups in this category are $\mathrm{GL}(d/e, q^e).e$ where e is a prime dividing d.

Assume that the FG-module M is irreducible. Holt & Rees [45] describe an extension of the MEATAXE to determine the centralising field E of M together with a $d \times d$ matrix which generates E as a field over F. In particular, M is absolutely irreducible if and only if $E = F$.

Holt *et al.* [46] present an algorithm to decide if an absolutely irreducible group acts semilinearly. In summary, we construct a subset S of random elements of the derived group of G, and now apply SMASH to decide if G preserves the appropriate decomposition with respect to $\langle S \rangle^G = G'$.

If G is both imprimitive and semilinear, we may fail to decide that G is semilinear, since repeated calls to SMASH always conclude that G acts imprimitively.

An implementation of the algorithm is distributed with MAGMA.

4.5. Tensor products.

Groups in this category preserve on V the structure of a tensor product of two subspaces, and maximal subgroups in this category are subgroups of the central product $\mathrm{GL}(e, q) \circ \mathrm{GL}(f, q)$ where $d = ef$.

Leedham-Green & O'Brien [58] provide a description of a tensor decomposition of V in terms of a projective geometry whose flats are certain subspaces of V. In [59] we exploit this geometrical approach and some other ideas to obtain a practical algorithm to decide tensor decomposability.

Here we summarise the approach, first recalling the concept of equivalence of tensor decompositions.

Definition 4.1. A u-tensor decomposition of V is a linear isomorphism α from $U \otimes W$ onto V, where U and W are vector spaces, with U of dimension u. If $\alpha : U \otimes W \to V$ and $\beta : U' \otimes W' \to V$ are u-tensor decompositions of V, then α and β are *equivalent* if there are linear isomorphisms $\phi : U \to U'$ and $\psi : W \to W'$ such that $\alpha = (\phi \otimes \psi)\beta$.

If V is an FG-module, where F is the underlying field and G is a group, then a u-tensor decomposition of V as FG-module requires U and W as above to be FG-modules, and α to be an FG-isomorphism; and in the definition of equivalence ϕ and ψ are required to be FG-isomorphisms.

A u-projective geometry on V, where u divides the dimension of V, is a projective geometry where the k-flats are of dimension ku, the join of two flats is their

sum, and their meet is their intersection. Thus, in a u-tensor decomposition of V, the subspaces of V that are the images of subspaces of $U \otimes W$ of the form $U \otimes W_0$, where W_0 runs through the set of subspaces of W, form a u-projective geometry on V. More generally, a u-tensor decomposition of V as FG-module gives rise to a u-projective geometry on V where W_0 runs through the set of FG-submodules of W. This projective geometry is G-invariant, in that the set of flats is G-invariant.

In [58] it was shown that this construction of a u-projective geometry from a tensor decomposition of V as FG-module sets up a one-to-one correspondence between the set of G-invariant projective geometries on V and the set of equivalence classes of tensor decompositions of V as FG-module. A *point* in the projective geometry corresponding to a u-tensor decomposition of V has dimension u as a subspace.

The following theorem is proved in [58].

Theorem 4.2. *Let V be a vector space of dimension uw. For each u-tensor decomposition $\alpha : U \otimes W \mapsto V$, define $\mathcal{F}(\alpha)$ to be $\{\alpha(U \otimes X) : X \leq W\}$. Then the map $[\alpha] \longmapsto \mathcal{F}(\alpha)$ is a bijection between the set of equivalence classes $[\alpha]$ of u-tensor decompositions of V and the set of u-projective geometries on V.*

In [58] an algorithm FindPoint having complexity $O(d^3)$ is presented: given as input a subspace $\mathcal{F}$ of V, it determines whether or not $\mathcal{F}$ is a flat in a G-invariant u-projective geometry on V, and, in the affirmative case, returns the corresponding tensor decomposition of V.

Hence the problem of finding a tensor decomposition of an FG-module V as $U \otimes W$, where U and W are modules for a covering group of G, is equivalent to constructing a point in one of the two corresponding projective geometries: a subspace of V of the form $u \otimes W$ or $U \otimes w$ for $u \in U \setminus \{0\}$ or $w \in W \setminus \{0\}$.

We use two approaches to find a flat in a suitable G-invariant projective geometry, or to prove that no such geometry exists.

If G does not act faithfully modulo scalars on one of the factors in the putative tensor decomposition, then (a variation of) Smash constructs the decomposition. If G acts faithfully modulo scalars on each of the factors in every tensor decomposition of V, then we consider the H-submodule structure of V for "suitable" subgroups H of G. A subgroup H is suitable if it is guaranteed to act reducibly on at least one of the tensor factors, say W, in every putative tensor decomposition. Then at least one of the H-invariant subspaces of V is a non-trivial flat in the corresponding u-projective geometry.

Hence, to apply the algorithm successfully, we wish to construct $H \leq G$ that normalises sufficiently few subspaces of V that we can process these subspaces, but which also acts reducibly on W if the required tensor factorisation exists. One simple criterion we employ is the following. If p is the characteristic of F and H is a p-local subgroup, then H cannot act irreducibly in any dimension greater than one: the subspace of V centralised by a p-group must be non-trivial, and this space is normalised by H.

Of course no suitable H may exist and hence the algorithm may fail to complete; our experience suggests that it is easy to construct the tensor decomposition, but

sometimes difficult to prove that no decomposition exists. An implementation of the algorithm is distributed with MAGMA.

4.6. Smaller field modulo scalars.

Let $G = \langle X \rangle$ be an absolutely irreducible subgroup of $\mathrm{GL}(d, K)$, and let F be a proper subfield of the finite field K.

Glasby, Leedham-Green & O'Brien [40] present an algorithm to decide constructively whether or not G is conjugate to a subgroup of $\mathrm{GL}(d, F).K^\times$, where $K^\times$ denotes the centre of $\mathrm{GL}(d, K)$.

Theorem 4.3. *There is a Las Vegas algorithm that takes as input the finite fields $F < K$, and an absolutely irreducible group $G := \langle X \rangle \leq \mathrm{GL}(d, K)$, and decides in $O(|X|d^3)$ field operations in K, plus $O^\sim(d \log q)$ field operations in F, whether or not G is conjugate to a subgroup of $\mathrm{GL}(d, F)$. If so, then a conjugating matrix is returned; otherwise* false *is returned.*

The algorithm of Glasby & Howlett [39] has similar complexity but assumes a discrete logarithm oracle for F. Our algorithm avoids use of the discrete logarithm, and hence its performance is demonstrably better if F is "large".

A variation of Theorem 4.3 allows us to decide membership in the Aschbacher category.

Theorem 4.4. *There is a Las Vegas algorithm that takes the same input as the algorithm in Theorem* 4.3*, but with the additional assumption that G' acts absolutely irreducibly on the given KG-module V; if G is conjugate to a subgroup of $\mathrm{GL}(d, F)K^\times$, it returns a conjugating matrix, or otherwise returns* false*. This algorithm has the same complexity as the algorithm in Theorem* 4.3*.*

We also generalise the algorithm of Theorem 4.4 in two ways to address the case when G acts absolutely irreducibly, but G' does not. It suffices, for the algorithm of Theorem 4.3 to produce a positive answer, that we find for each $g \in X$ a scalar $k_g \in K^\times$ such that if g is replaced by $k_g g$ then the resulting set generates a group that can be conjugated into $\mathrm{GL}(d, F)$. Thus we find such scalars by considering the elements of X in turn, and then carry out a backtrack search through all possible scalars; in practice we restrict the choice of scalars significantly. The second approach is to use Clifford's theorem [30] to analyse the structure of the KG-module.

An implementation of the algorithm is distributed with MAGMA.

4.7. Normalisers of p-groups.

Groups in this category are the normalisers of certain absolutely irreducible, symplectic-type r-groups, where r is a prime, d a power of r and $q \equiv 1 \pmod{r}$.

Niemeyer [69] proved the following.

Theorem 4.5. *Let p and r be primes with $r \geq 3$. Let e be the smallest integer such that $p^e \equiv 1 \bmod r$ and put $q = p^e$. Let R_0 be a given embedding of a symplectic-type extraspecial r-subgroup R of order r^3 and exponent r into $\mathrm{GL}(r, q)$. There is a constructive, one-sided Monte Carlo algorithm which takes as input a group G generated by a set X of matrices in $\mathrm{GL}(r, q)$ and decides whether or not G has a normal subgroup isomorphic to R_0. The algorithm costs $O(\xi + (\log r \log\log r + \log q + |X|)\mu + \delta)$ field operations, where μ is the cost of a group operation, ξ is the cost of selecting a random element, and δ is the cost of finding an r-th root of an element in $\mathrm{GF}(q)$.*

The general case is considered by Brooksbank, Niemeyer & Seress [23]. Implementations of these algorithms are available in GAP and Magma.

4.8. Tensor-induced groups.

Let $G \leq \mathrm{GL}(d, q)$ be tensor-induced. Then G preserves a decomposition of V as

$$U_1 \otimes U_2 \otimes \cdots \otimes U_r$$

where each U_i has dimension $u > 1$ and $r > 1$, and the set of U_i is permuted by G.

Leedham-Green & O'Brien [60] present an algorithm to decide if G is tensor-induced. We may readily reduce to the case where G acts primitively on the set of tensor factors. In summary, we consider homomorphisms from G onto a *primitive* subgroup of S_r, and construct such mappings, or prove that none exists.

In particular, we construct a set of subsets of G in one-to-one correspondence with the set of conjugacy classes of subgroups of G of index r, each subset generating a group in the corresponding class.

The standard low-index subgroup algorithm described in [80] constructs such classes when G is a finitely-presented group. Critically, the relations used to obtain subgroups of index at most r do not need to be satisfied by G, but rather by G/K where K is a normal subgroup contained in the intersection of the kernels of all homomorphisms of G into S_r.

We construct a generating set for a subgroup K of K_r, the verbal subgroup of G corresponding to the variety generated by S_r, by evaluating instances of some known laws of the variety. This we can do modulo the assumption that r is *small*. (This is a realistic assumption: if we assume that $d \leq 500$, then $r \leq 5$, unless $u = 2$, in which case $r \leq 8$.)

We next obtain a presentation for a preimage of G/K; here we use the algorithm of [60] to construct random elements of a normal subgroup, and the algorithm outlined in Section 2 to estimate the order of an element of G modulo a normal subgroup.

We apply the low-index subgroup algorithm to this presentation to construct subgroups of bounded index and obtain their preimages in G.

We next determine whether or not a subgroup M of appropriate index in G preserves a tensor decomposition of V with factors U of dimension u and W of

dimension u^{r-1}. If M does *not* preserve such a tensor decomposition, then G is not tensor-induced and the algorithm terminates.

If M preserves such a tensor decomposition, it remains to decide whether or not G is tensor-induced from a subgroup of index r. In particular, we determine whether or not W can be decomposed into $r-1$ tensor factors of dimension u in such a way that the resulting set of r u-dimensional tensor factors of V is permuted by G.

An implementation of the algorithm is distributed with MAGMA.

5. Exploiting the geometry

In ongoing work, Leedham-Green and O'Brien have developed the concept of a *composition tree*, which seeks to realise and exploit the Aschbacher classification. Leedham-Green [57] provides a detailed description of this concept and its practical realisation. Here we summarise it briefly.

A composition series for a group R can be viewed as a labelled rooted binary tree. The nodes correspond to sections of R, the root node to R. A node that corresponds to a section K of R, and is not a leaf, has a left descendant corresponding to a proper normal subgroup N of K and a right descendant corresponding to K/N. The right descendant is an image under a homomorphism; usually these arise naturally from the Aschbacher category of the group, but we also exploit additional ones applying to unipotent and soluble groups. The left descendant of a node is the kernel of the chosen homomorphism.

The tree is constructed in *right depth-first order*. Namely, we process the node associated with K: if K is not a leaf, construct recursively the subtree rooted at its right descendant I, then the subtree rooted at its left descendant N.

It is easy to construct I, since it is the image of K under a homomorphism ϕ. We generate a random element of N as follows. Let $K = \langle x_1, \ldots, x_m \rangle$, and let $I = \phi(K) = \langle \overline{x_1}, \ldots, \overline{x_m} \rangle$. Choose random $k \in K$, and evaluate $\phi(k) \in I$. By solving the word-problem for I, we establish that $\phi(k) = w(\overline{x_1}, \ldots, \overline{x_m})$. Then the *residue* $k \cdot w(x_1, \ldots, x_m)^{-1} \in N$. Hence, by selecting sufficient random elements of K, we construct with high probability a generating set for N.

We assign to the root node R a set of random elements which are used for "quality control" in constructing the composition tree. Their images and residues are determined for each new node constructed. We test if these random elements satisfy the homomorphism specified; if their images under the homomorphism are in the image; if the residues are in the kernel. If any of these tests fail, we know that the generating set for some *kernel* which is an ancestor of the node is not correct. We add more generators to this kernel and construct the subtree having this root again.

We solve the word-problem *directly* for a *leaf* – namely, a composition factor of the root group R – using a variety of techniques which we survey in Section 7. If we solve the word-problem for the left and right descendants of a node, then we

readily solve the word-problem for the node, and so recursively obtain a solution for the root node. Hence, given $x \in \mathrm{GL}(d,q)$, we can decide if $x \in R$; if so, we can write x as a word in the user-supplied defining generators of R.

Recently, Mark Stather refined the composition tree concept to construct a *chief tree* of a group, whose leaves are the chief factors of the group.

6. Non-constructive recognition

The algorithms to name a finite simple group exploit the concept of a *primitive prime divisor*.

Let b, e be positive integers with $b > 1$. A prime r dividing $b^e - 1$ is a primitive prime divisor of $b^e - 1$ if $r | (b^e - 1)$ but $r \not| (b^i - 1)$ for $1 \leq i < e$. Zsigmondy [87] proved that $b^e - 1$ has a primitive prime divisor unless $(b, e) = (2, 6)$ or $e = 2$ and $b + 1$ is a power of 2. Recall that

$$|\mathrm{GL}(d,q)| = q^{\binom{d}{2}} \prod_{i=1}^{d} (q^i - 1).$$

Hence primitive prime divisors of $q^e - 1$ for various $e \leq d$ divide both the orders of $\mathrm{GL}(d,q)$ and of classical groups.

We say that $g \in \mathrm{GL}(d,q)$ is a *ppd*$(d,q;e)$-element (or sometimes simply a *ppd-element*) if its order is divisible by some primitive prime divisor of $q^e - 1$.

6.1. Classical groups in natural representation.

Much of the recent activity on algorithms for linear groups was stimulated by Neumann & Praeger [68], who presented a Monte Carlo algorithm to decide whether or not a subgroup of $\mathrm{GL}(d,q)$ contains $\mathrm{SL}(d,q)$.

Niemeyer & Praeger [71] answer the equivalent question for an arbitrary classical group. Underpinning the work is a classification of the subgroups of $\mathrm{GL}(d,q)$ containing ppd-elements for $e > d/2$ obtained by Guralnick *et al.* [41]. In [71], they refine this classification, focusing on pairs of elements in G which are $\mathrm{ppd}(d,q;e_1)$ and $\mathrm{ppd}(d,q;e_2)$ for $d/2 < e_1 < e_2 \leq d$. With few exceptions, if G contains such elements, then G contains one of the classical groups. They determine the proportion of such ppd-elements in classical groups, and also list the exceptions. In summary, the resulting Monte Carlo algorithms are highly efficient, having complexity $O(\log\log d(\xi + d^\omega (\log q)^2))$, where ξ is the cost of selecting a random element and d^ω is the cost of matrix multiplication.

For an excellent account of this and related work, see Praeger [76]. For a report on the resulting implementation, which is distributed with Magma, see [70].

6.2. Black-box groups of Lie type.

Babai *et al.* [8] present a black-box algorithm to name a group G of Lie type in known defining characteristic p. The algorithm selects a sample of random elements in G, and determines whether the orders of these elements are divisible by certain primitive prime divisors. From this divisibility information, it constructs the *Artin invariants* of G: the leading invariant is usually the largest k such that G contains elements of order $\mathrm{ppd}(p,k)$-order. With certain exceptions, the Artin invariants determine G. The algorithm of Altseimer & Borovik [1] distinguishes between $\mathrm{P}\Omega(2m+1,q)$ and $\mathrm{PSp}(2m,q)$ for odd $q>3$.

The central result of [8] is the following.

Theorem 6.1. *Given a black-box group G isomorphic to a simple group of Lie type of known characteristic, the standard name of G can be computed using a polynomial-time Monte Carlo algorithm.*

In 2001 Malle and O'Brien developed a practical implementation of the resulting algorithm. Our procedure takes as input a quasisimple group in known defining characteristic. We also include identification procedures for the other quasisimple groups. If the non-abelian composition factor is alternating or sporadic, then we identify it by considering the orders of random elements. Our implementation is distributed with GAP and Magma.

Observe that Theorem 6.1 assumes that the defining characteristic of the input group of Lie type is *known*. The algorithm of Kantor & Seress [54] to determine the characteristic does not appear to be practical; an alternative was developed by Liebeck & O'Brien [63] and our implementation is distributed with Magma.

7. Solving the word-problem

We focus on approaches which solve the word-problem – and sometimes provide much additional information – for simple groups.

7.1. Black-box classical groups.

Cooperman, Finkelstein & Linton [36] made a critical breakthrough, presenting a constructive recognition algorithm for $\mathrm{GL}(n,2)$.

This inspired the work of Kantor & Seress [53]; in summary, they prove the following.

Theorem 7.1. *There is a Las Vegas algorithm which, when given as input a black-box perfect group $G \leq \mathrm{GL}(d,q)$ where $G/Z(G)$ is isomorphic to a classical simple group C of known characteristic, produces a constructive isomorphism $G/Z \longmapsto C$.*

A partial implementation of the algorithm, developed by Brooksbank, Seress and others, is available in GAP and Magma. The algorithm is not polynomial in the size of input: its running time has a factor of $q = p^e$ because a necessary step is to find an element of order p.

Recall that $g \in G$ is *p-singular* if its order is divisible by p. A group of Lie type having defining characteristic p has a small proportion of p-singular elements. Combining the results of Isaacs, Kantor & Spaltenstein [51] and Guralnick & Lübeck [42], we obtain the following.

Theorem 7.2. *If G is a group of Lie type defined over $\mathrm{GF}(q)$, then $\frac{2}{5q} < \rho(G) < \frac{5}{q}$, where $\rho(G)$ denotes the proportion of p-singular elements in G.*

Brooksbank & Kantor [22] identify that the obstruction to a polynomial-time algorithm for constructive recognition of the classical groups is $\mathrm{PSL}(2, q)$. Babai & Beals [7] formulate the problem explicitly as follows.

Problem 7.3. *Find an element of order p in $\mathrm{PSL}(2, p^e)$ as a word in its defining generators in polynomial time.*

Since $\rho(\mathrm{PSL}(2, q)) \leq 2/q$, a random search will involve $O(q)$ selections.

A consequence of the work of Landazuri & Seitz [56] is that the degree of a faithful projective representation of $\mathrm{PSL}(2, q)$ in cross characteristic is polynomial in q rather than in $\log q$. Hence the critical case is a matrix representation of $\mathrm{SL}(2, q)$ in defining characteristic.

Conder & Leedham-Green [32] and Conder, Leedham-Green & O'Brien [33] present an algorithm which constructively recognises $\mathrm{SL}(2, q)$ as a linear group in defining characteristic in time polynomial in the size of the input. The principal result is the following.

Theorem 7.4. *Let G be a subgroup of $\mathrm{GL}(d, F)$ for $d \geq 2$, where F is a finite field of the same characteristic as $\mathrm{GF}(q)$; assume that G is isomorphic modulo scalars to $\mathrm{PSL}(2, q)$. Then, subject to a fixed number of calls to a discrete log oracle for $\mathrm{GF}(q)$, there is a Las Vegas algorithm that constructs an epimorphism from G to $\mathrm{PSL}(2, q)$ at a cost of at most $O(d^5\tau(d))$ field operations, where $\tau(d)$ denotes the number of divisors of d.*

Underpinning our work is a well-known characterisation of the absolutely irreducible representations of $\mathrm{SL}(2, q)$, due to Brauer & Nesbitt [15].

Theorem 7.5. *Let K be a finite field of characteristic p, and let V be an absolutely irreducible KG-module for $G = \mathrm{SL}(2, q)$, where $q = p^e$. Suppose that V cannot be written over a smaller field. Then K is a subfield of $\mathrm{GF}(q)$, and $V \otimes_K \mathrm{GF}(q) \simeq T_1 \otimes T_2 \otimes \cdots \otimes T_t$, where T_i is the s_i-fold symmetric power S_i of the natural $\mathrm{GF}(q)[G]$-module M twisted by the f_i th power of the Frobenius map, with $0 \leq f_1 < f_2 < \cdots < f_t < e$, and $1 \leq s_i < p$ for all i.*

Let q be a power of a prime p, and let V be a finite-dimensional vector space over a finite field of characteristic p. In summary, our algorithm takes as input a subset X of the linear group $\mathrm{GL}(V)$ that generates a group G isomorphic to $\mathrm{SL}(2,q)$ or to $\mathrm{PSL}(2,q)$, and constructs the natural projective representation of G by constructing the image of X under a homomorphism of G onto $\mathrm{PSL}(2,q)$.

How do we find a transvection in the natural representation H of $\mathrm{SL}(2,q)$? We find by random search an element a of order $q-1$ in H, and a random conjugate b of a. Next we construct $c \in H$ and an integer i such that $b^i c$ and a have a common eigenvector. Observe that $[a, b^i c]$ is a transvection. While a suitable c can be found easily, computing i relies on a discrete logarithm oracle.

Brooksbank [19], [21] and Brooksbank & Kantor [22] have exploited this work to produce better constructive recognition algorithms for black-box classical groups. Kantor & Seress [55] summarise the outcome as follows.

Theorem 7.6. *There is a Monte Carlo algorithm which, when given as input a black-box G such that $C = G/Z(G)$ is $\mathrm{PSL}(d,q)$, $\mathrm{PSp}(2m,q)$ or $\mathrm{PSU}(d,q)$ and a constructive recognition oracle for $\mathrm{SL}(2,q)$, outputs a constructive isomorphism $G/Z(G) \longmapsto C$. The running time of the resulting algorithms is a polynomial in the input length plus the time of polynomially many calls to the $\mathrm{SL}(2,q)$ oracle.*

For example, the complexity of Brooksbank's algorithm [21] for $\mathrm{PSU}(d,q)$ is $O(d^2 \log d(\xi + \chi \log q + d \log^4 q)$, where ξ is the cost of selecting a random element and χ is the cost of an $\mathrm{SL}(2,q)$-oracle.

Recently Brooksbank & Kantor [24] developed an algorithm having similar complexity for the orthogonal groups.

7.2. Classical groups in their natural representation.

The algorithm of Celler & Leedham-Green [29] for constructive recognition of $\mathrm{SL}(d,q)$ in its natural representation has effective cost $O(d^4 q)$. Recently, Brooksbank [20] developed similar algorithms for other classical groups in their natural representation: their effective cost is $O(d^5 \log^2 q)$, subject to calls to an $\mathrm{SL}(2,q)$ oracle.

In ongoing work, Leedham-Green and O'Brien are developing new algorithms for the classical groups, given as linear groups in defining characteristic; these use an $\mathrm{SL}(2,q)$ oracle and their complexity involves $\log q$.

7.3. Alternating groups.

Beals *et al.* [11] prove the following.

Theorem 7.7. *Black-box groups isomorphic to A_n or S_n with known value of n can be recognised constructively, in $O(\xi n + \mu |X| n \log n)$ time, where ξ is the time to construct a random element, μ is the time for a group operation, and X is the input generating set for the group.*

Beals *et al.* [12] present an alternative linear group algorithm designed for the deleted permutation module. Implementations of these algorithms are available in GAP and Magma.

An alternative algorithm, developed by Bratus & Pak [16], was further refined and implemented in Magma by Derek Holt.

7.4. Using centralisers of involutions.

The centraliser of an involution in a black-box group having an order oracle can be constructed using an algorithm of Bray [17].

Assume we wish to construct elements of $C_G(h)$, for involution $h \in G$. Construct a conjugate h^k of h, where k is a random element of G. Let D be the dihedral group generated by h and h^k, and let the order of D be $2n$.

(i) If n is odd, D contains an element t such that $h^t = h^k$. Then tk^{-1} is an element of $C_G(h)$.

(ii) If n is even, D contains a central involution x. Then x and $x^{k^{-1}}$ both centralise h.

It is easy to prove that the elements of $C_G(h)$ produced under step (i) are uniformly distributed. Parker & Wilson [75] prove that certain classical groups contain "sufficient" elements of this type having odd order.

Theorem 7.8. *There is an absolute constant c such that if G is a finite simple classical group, with natural module of dimension d over a field of odd characteristic, and h is an involution in G, then $[h, g]$ has odd order for at least a proportion c/d of the elements $g \in G$.*

Borovik [13] considers involution centralisers in the study of black-box groups and announced a weaker version of this theorem. A result similar to Theorem 7.8 is also established for the exceptional groups in [75].

For each sporadic group we can calculate explicitly the proportion of $[h, g]$ which have odd order. Since, for every class of involutions, this proportion is at least 17%, we can readily construct centralisers.

The *centraliser-of-involution* algorithm [44] reduces the problem of testing whether an arbitrary $g \in G$ is a member of $H \leq G$ to instances of the same problem for $C_H(t)$ for (at most) three involutions $t \in H$. The algorithm is constructive: if $g \in H$ then it returns a word for g in the generators of H.

We summarise the algorithm. Assume we are given a black-box group G with an order oracle, $g \in G$, and a subgroup H of G. We wish to decide whether or not $g \in H$.

1. Find $h \in H$ such that $|gh| = 2\ell$. Now define $z = (gh)^\ell$.

2. Find x, an H-involution, such that $|xz| = 2m$. Now define $y = (xz)^m$.

3. Construct $X = C_H(x)$ and decide if $y \in X$.

4. If so, construct $Y = C_H(y)$ and decide if $z \in Y$.

5. If so, construct $Z = C_H(z)$ and decide if $gh \in Z$.

Note that $\langle x, z\rangle$ is D_{2m} having central involution $y = (xz)^m$. Hence y is in the centraliser of x and z is in the centraliser of y.

If any of the membership tests fail, we immediately conclude that $g \notin H$; otherwise, on termination, we have proved that $g \in H$.

An implementation is distributed with MAGMA.

7.5. The Schreier-Sims approach.

Underpinning most effective algorithms for permutation groups is the concept of a *base and strong generating set* (BSGS).

Let a group G act faithfully on $\Omega = \{1, \ldots, n\}$. Recall that a *base* for G is a sequence of points $B = [\beta_1, \beta_2, \ldots, \beta_k]$ such that the sequence stabiliser $G_{\beta_1,\beta_2,\ldots,\beta_k} = 1$. This structure determines a chain of stabilisers

$$G = G^{(1)} \geq G^{(2)} \geq \cdots \geq G^{(k)} \geq G^{(k+1)} = 1,$$

where $G^{(i)} = G_{\beta_1,\beta_2,\ldots,\beta_{i-1}}$. A *strong generating set* corresponding to B is a subset S of G such that $G^{(i)} = \left\langle S \cap G^{(i)} \right\rangle$, for $i = 1, \ldots, k$.

The central task is the construction of *basic orbits* – the orbit B_i of the base point β_{i+1} under $G^{(i)}$. Observe that $|G^{(i)} : G^{(i+1)}| = |B_i|$, a *basic index*. Using Schreier's Lemma, Sims [79] presented a deterministic algorithm to construct the required strong generating sets. For an analysis of the algorithm, see Seress [77, p. 64].

By contrast, the random Schreier-Sims, introduced by Leon [61], finds generating sets by considering random elements of G. It is usually significantly faster and provides smaller strong generating sets. In practice, it terminates when some *stopping condition* becomes true. Usually, we stop when a predetermined number, N, of consecutive random elements have all been found to be redundant as strong generators. If the random elements are uniformly distributed, the probability that we do not have a complete BSGS is now less than 2^{-N}. If the order of G is known in advance, we can terminate when the product of basic indices reaches this value.

Of course, there is a natural faithful action of a linear group $G \leq \mathrm{GL}(d, q)$ on the underlying vector space $V = \mathrm{GF}(q)^d$: namely, $v^g = v \cdot g$ for $v \in V$ and $g \in G$. Hence we can apply the Schreier-Sims algorithm to G and construct a BSGS for its action on the vectors of V, where the base points are standard basis vectors for V. Observe however that the size of V is q^d and so grows *exponentially* with d. The basic orbits obtained are usually very large; if G is a simple group, the first basic index is often $|G|$.

By choosing base points which give shorter basic orbits, we extend significantly the range of application of the Schreier-Sims. Butler [25] first developed

the Schreier-Sims algorithm for linear groups, choosing as base points the one-dimensional subspaces of V. Murray & O'Brien [67] developed a more general strategy for selecting base points for linear groups which we expect *a priori* to have "small" orbits. In summary, we select some common eigenvectors for a collection of random elements of the group, and use related spaces to obtain a base.

Most critical to the successful application of the Schreier-Sims algorithm is the index $|G^{(i)}:G^{(i+1)}|$. While S_n has a subgroup of index n, the "optimal" subgroup chain for $\mathrm{GL}(d,q)$ is

$$\mathrm{GL}(d,q) \geq q^{d-1}.\mathrm{GL}(d-1,q) \geq \mathrm{GL}(d-1,q) \geq \ldots$$

where the leading index is $q^d - 1$ and so grows exponentially with d. Further, many linear groups have no "small-degree" permutation representation and so no useful stabiliser-chain. For example, the largest maximal subgroup of the sporadic simple group J_4 has index 173 067 389.

Despite these limitations, the algorithms underpin most of the long-standing machinery for computing with linear groups. Implementations are available in GAP and Magma, and are very effective for "small" degree representations defined over "small" fields. While the model borrows heavily from permutation groups, it does not write down an explicit permutation representation for the group, relying instead on a stabiliser-chain. See, for example, the algorithm of Butler & Cannon [26] to construct centralisers of elements of linear groups.

An algorithm which uses subset chains to solve the word-problem for black-box groups is described by Ambrose *et al.* [2].

7.6. Sporadic groups.

Wilson [84] introduced the concept of *standard generators* for the sporadic groups. He and others provide black-box algorithms for their construction. Generating sets for maximal subgroups, representative of conjugacy classes and other structural information are now obtained by evaluating known words in these standard generators. For further details, see the Atlas WEB site [85].

For each sporadic group, O'Brien & Wilson [72] present black-box algorithms which construct chains of subgroups. For a specific matrix representation, each chain now determines a stabiliser chain for (variations of) the Schreier-Sims algorithm. Some subgroups in the chain act reducibly on the underlying vector space; hence we construct a module composition series, and, by estimating orbit sizes, select "good" base points for the Schreier-Sims algorithm. With this assistance, the Schreier-Sims or the centraliser-of-involution algorithm [44] solves the word-problem for all Atlas representations [85] of most sporadic groups; the exceptions are the Baby Monster and the Monster where strategies developed by Wilson and others are employed [86]. Implementations are available in Magma.

8. Presentations for groups

Most of the algorithms surveyed here are randomised; Monte Carlo or Las Vegas in nature, they rely on random selections.

How do we verify the results obtained? For example, how do we prove that the composition tree for a given group G is correct? One method of verification is to use a presentation. By constructing a composition tree for G, we obtain M, a group with composition factors the leaves. Then $|M| \leq |G|$, perhaps *properly* if we fail to construct completely the kernel of a homomorphism. We now construct a presentation for M and verify that G satisfies the relations for M. Hence G is a quotient of M and we conclude that $G = M$.

Since we must evaluate relations, we are interested in "short" presentations. The *length* of a presentation is the number of symbols needed to write the presentation. A presentation for G is *short* if its length is $O(\log^2 |G|)$.

Combining the results of Babai *et al.* [6], Hulpke & Seress [49], and Suzuki [83], we obtain the following.

Theorem 8.1. *For every finite simple group except ${}^2G_2(q)$ there is a known short presentation.*

For Lie rank at least 2, these are reduced versions of the Curtis-Steinberg-Tits presentations.

Conder, Leedham-Green & O'Brien [34] prove the following.

Theorem 8.2. *The alternating and symmetric groups of degree n have presentations on $\log n$ generators, where the number of relators is $O(\log n)$, and the presentation length is $O(\log n \log\log n)$.*

This represents a significant improvement over known (Coxeter) presentations which have length $O(n^2)$. The consequent shorter presentations for the classical groups are described in [35].

References

[1] Christine Altseimer and Alexandre V. Borovik. Probabilistic recognition of orthogonal and symplectic groups. In *Groups and Computation, III (Columbus, OH, 1999)*, volume 8 of *Ohio State Univ. Math. Res. Inst. Publ.*, pages 1–20. de Gruyter, Berlin, 2001.

[2] Sophie Ambrose, Max Neunhöffer, Cheryl E. Praeger and Csaba Schneider. Generalised sifting in black-box groups. *LMS J. Comput. Math.*, 8:217-250, 2005.

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

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

[5] L. Babai. Randomization in group algorithms: conceptual questions. In *Groups and Computation, II (New Brunswick, NJ, 1995)*, 1–17, Amer. Math. Soc., Providence, RI, 1–17, 1997.

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

[7] László Babai and Robert Beals. A polynomial-time theory of black box groups. I. In *Groups St. Andrews 1997 in Bath, I*, volume 260 of *London Math. Soc. Lecture Note Ser.*, pages 30–64, Cambridge, 1999. Cambridge Univ. Press.

[8] László Babai, William M. Kantor, Péter P. Pálfy, and Ákos Seress. Black-box recognition of finite simple groups of Lie type by statistics of element orders. *J. Group Theory*, 5(4):383–401, 2002.

[9] László Babai and Aner Shalev. Recognizing simplicity of black-box groups and the frequency of p-singular elements in affine groups. In *Groups and Computation* III, Ohio State Univ. Math. Res. Inst. Publ., pages 39–62. de Gruyter, Berlin, 2001.

[10] László Babai and Endre Szemerédi. On the complexity of matrix group problems, I. In *Proc. 25th IEEE Sympos. Foundations Comp. Sci.*, pages 229–240, 1984.

[11] Robert Beals, Charles R. Leedham-Green, Alice C. Niemeyer, Cheryl E. Praeger, and Ákos Seress. A black-box group algorithm for recognizing finite symmetric and alternating groups. I. *Trans. Amer. Math. Soc.*, 355(5):2097–2113, 2003.

[12] Robert Beals, Charles R. Leedham-Green, Alice C. Niemeyer, Cheryl E. Praeger, and Ákos Seress. Constructive recognition of finite alternating and symmetric groups acting as matrix groups on their natural permutation modules. *J. Algebra*, 292:4–46, 2005.

[13] A.V. Borovik. Centralisers of involutions in black box groups. In *Computational and statistical group theory (Las Vegas, NV/Hoboken, NJ, 2001)*, 7–20, *Contemp. Math.*, 298, Amer. Math. Soc., Providence, RI, 2002.

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

[15] R. Brauer and C. Nesbitt. On the modular characters of groups, *Ann. of Math.* 42:556–590, 1941.

[16] Sergey Bratus and Igor Pak. Fast constructive recognition of a black box group isomorphic to S_n or A_n using Goldbach's conjecture. *J. Symbolic Comput.* 29:33–57, 2000.

[17] John N. Bray. An improved method for generating the centralizer of an involution. *Arch. Math. (Basel)*, 74:241–245, 2000.

[18] John Brillhart, D.H. Lehmer, J.L. Selfridge, Bryant Tuckerman, and S.S. Wagstaff, Jr. *Factorizations of* $b^n \pm 1$, volume 22 of *Contemporary Mathematics*. American Mathematical Society, Providence, RI, second edition, 1988. http://www.cerias.purdue.edu/homes/ssw/cun/index.html.

[19] Peter A. Brooksbank. A constructive recognition algorithm for the matrix group $\Omega(d, q)$. In *Groups and Computation, III (Columbus, OH, 1999)*, volume 8 of *Ohio State Univ. Math. Res. Inst. Publ.*, pages 79–93. de Gruyter, Berlin, 2001.

[20] Peter A. Brooksbank. Constructive recognition of classical groups in their natural representation. *J. Symbolic Comput.*, 35:195–239, 2003.

[21] Peter A. Brooksbank. Fast constructive recognition of black-box unitary groups. *LMS J. Comput. Math.*, 6:162–197 (electronic), 2003.

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

[23] Peter Brooksbank, Alice C. Niemeyer and Ákos Seress. A reduction algorithm for matrix groups with an extraspecial normal subgroup. These Proceedings.

[24] Peter A. Brooksbank and William M. Kantor. Fast constructive recognition of black box orthogonal groups. *J. Algebra*, 2006.

[25] Gregory Butler. The Schreier algorithm for matrix groups. In SYMSAC '76, *Proc. ACM Sympos. symbolic and algebraic computation*, pages 167–170, New York, 1976. (New York, 1976), Association for Computing Machinery.

[26] Gregory Butler and John J. Cannon. Computing in permutation and matrix groups I: Normal closure, commutator subgroups, series. *Math. Comp.*, 39:663–670, 1982.

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

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

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

[30] A.H. Clifford. Representations induced in an invariant subgroup. *Ann. of Math.*, 38:533–550, 1937.

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

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

[33] M.D.E. Conder, C.R. Leedham-Green, and E.A. O'Brien. Constructive recognition of PSL$(2, q)$. *Trans. Amer. Math. Soc.*, 358:1203-1221, 2006.

[34] M.D.E. Conder, C.R. Leedham-Green, and E.A. O'Brien. Short presentations for alternating and symmetric groups. Preprint, 2005.

[35] M.D.E. Conder, C.R. Leedham-Green, and E.A. O'Brien. Short presentations for classical groups. Preprint, 2005.

[36] G. Cooperman, L. Finkelstein, and S. Linton. Constructive recognition of a black-box group isomorphic to GL$(n, 2)$. In *Groups and Computation II*, volume 28 of *Amer. Math. Soc. DIMACS Series*, pages 85–100. (DIMACS, 1995), 1997.

[37] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *J. Symbolic Comput.* 9:251–280, 1990.

[38] The GAP Group. GAP – Groups, Algorithms, and Programming, Version 4.4; 2004. (http://www.gap-system.org)

[39] S.P. Glasby and R.B. Howlett. Writing representations over minimal fields, *Comm. Algebra* 25:1703–1712, 1997.

[40] S.P. Glasby, C.R. Leedham-Green and E.A. O'Brien. Writing projective representations over subfields. *J. Algebra*, 295:51–61, 2006.

[41] Robert Guralnick, Tim Penttila, Cheryl E. Praeger, and Jan Saxl. Linear groups with orders having certain large prime divisors. *Proc. London Math. Soc.*, 78:167–214, 1997.

[42] R.M. Guralnick and F. Lübeck, On p-singular elements in Chevalley groups in characteristic p. In *Groups and Computation, III (Columbus, OH, 1999)*, volume 8 of *Ohio State Univ. Math. Res. Inst. Publ.*, pages 169–182, de Gruyter, Berlin, 2001.

[43] G. Hiss and G. Malle. Low-dimensional representations of quasi-simple groups. *LMS J. Comput. Math.*, 4:22–63, 2001. Also: Corrigenda *LMS J. Comput. Math.*, 5:95–126, 2002.

[44] P.E. Holmes, S.A. Linton, E.A. O'Brien, A.J.E. Ryba and R.A. Wilson. Constructive membership testing in black-box groups. Preprint, 2005.

[45] Derek F. Holt and Sarah Rees. Testing modules for irreducibility. *J. Austral. Math. Soc. Ser. A*, 57:1–16, 1994.

[46] Derek F. Holt, C.R. Leedham-Green, E.A. O'Brien, and Sarah Rees. Computing matrix group decompositions with respect to a normal subgroup. *J. Algebra*, 184:818–838, 1996.

[47] Derek F. Holt, C.R. Leedham-Green, E.A. O'Brien, and Sarah Rees. Testing matrix groups for primitivity. *J. Algebra*, 184:795–817, 1996.

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

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

[50] B. Huppert. *Endliche Gruppen I*, volume 134 of *Grundlehren Math. Wiss.* Springer-Verlag, Berlin, Heidelberg, New York, 1967.

[51] I.M. Isaacs, W.M. Kantor and N. Spaltenstein. On the probability that a group element is p-singular. *J. Algebra* 176:139–181, 1995.

[52] Gábor Ivanyos and Klaus Lux. Treating the exceptional cases of the MeatAxe. *Experiment. Math.*, 9:373–381, 2000.

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

[54] William M. Kantor and Ákos Seress. Prime power graphs for groups of Lie type. *J. Algebra*, 247(2):370–434, 2002.

[55] W.M. Kantor and Á. Seress. Computing with matrix groups. In *Groups, Combinatorics & Geometry (Durham, 2001)*, 123–137, World Sci. Publishing, River Edge, NJ, 2003.

[56] Vicente Landazuri and Gary M. Seitz. On the minimal degrees of projective representations of the finite Chevalley groups. *J. Algebra*, 32:418–443, 1974.

[57] C.R. Leedham-Green. The computational matrix group project. In *Groups and Computation, III (Columbus, OH, 1999)*, 229–248. de Gruyter, Berlin, 2001.

[58] C.R. Leedham-Green and E.A. O'Brien. Tensor products are projective geometries. *J. Algebra*, 189:514–528, 1997.

[59] C.R. Leedham-Green and E.A. O'Brien. Recognising tensor products of matrix groups. *Internat. J. Algebra Comput.*, 7:541–559, 1997.

[60] C.R. Leedham-Green and E.A. O'Brien. Recognising tensor-induced matrix groups. *J. Algebra*, 253:14–30, 2002.

[61] Jeffrey S. Leon. On an algorithm for finding a base and strong generating set for a group given by generating permutations. *Math. Comp.*, 20:941–974, 1980.

[62] Martin W. Liebeck. On the orders of maximal subgroups of the finite classical groups. *Proc. London Math. Soc.* (3), 50:426–446, 1985.

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

[64] F. Lübeck. Small degree representations of finite Chevalley groups in defining characteristic. *LMS J. Comput. Math.* 4: 135–169, (electronic), 2001.

[65] Eugene M. Luks. Computing in solvable matrix groups. In *Proc. 33rd IEEE Sympos. Foundations Comp. Sci.*, 111–120, 1992.

[66] T. Miyazaki. Deterministic algorithms for management of matrix groups. In *Groups and Computation, III (Columbus, OH, 1999)*, 265–280. de Gruyter, Berlin, 2001.

[67] Scott H. Murray and E.A. O'Brien. Selecting base points for the Schreier-Sims algorithm for matrix groups. *J. Symbolic Comput.*, 19:577–584, 1995.

[68] Peter M. Neumann and Cheryl E. Praeger. A recognition algorithm for special linear groups. *Proc. London Math. Soc.* (3), 65:555–603, 1992.

[69] Alice C. Niemeyer. Constructive recognition of normalisers of small extra-special matrix groups. *Internat. J. Algebra Comput.*, 15:367–394, 2005.

[70] Alice C. Niemeyer and Cheryl E. Praeger. Implementing a recognition algorithm for classical groups. In *Groups and Computation, II (New Brunswick, NJ, 1995)*, volume 28 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 273–296, Providence, RI, 1997. Amer. Math. Soc.

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

[72] E.A. O'Brien and R.A. Wilson. Optimal stabiliser chains for sporadic and other linear groups. Preprint, 2005.

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

[74] R.A. Parker. The computer calculation of modular characters (the Meat-Axe). In M.D. Atkinson, editor, *Computational Group Theory*, pages 267–274, London, New York, 1984. (Durham, 1982), Academic Press.

[75] Christopher W. Parker and Robert A. Wilson. Recognising simplicity of black-box groups. Preprint, 2005.

[76] Cheryl E. Praeger. Primitive prime divisor elements in finite classical groups. In *Groups St. Andrews 1997 in Bath, II*, 605–623, Cambridge Univ. Press, Cambridge, 1999.

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

[78] Igor E. Shparlinski, *Finite fields: theory and computation. The meeting point of number theory, computer science, coding theory and cryptography.* Mathematics and its Applications, 477. Kluwer Academic Publishers, Dordrecht, 1999.

[79] Charles C. Sims. Computational methods in the study of permutation groups. In *Computational problems in abstract algebra*, pages 169–183, Oxford, 1970. (Oxford, 1967), Pergamon Press.

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

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

[82] V. Strassen. Gaussian elimination is not optimal. *Numer. Math.* 13:354–356, 1969.

[83] Michio Suzuki. On a class of doubly transitive groups. *Ann. of Math. 2*, 75:105–145, 1962.

[84] Robert A. Wilson. Standard generators for sporadic simple groups. *J. Algebra*, 184(2):505–515, 1996.

[85] R.A. Wilson *et al.*. Atlas of Finite Group Representations, at http://brauer.maths.qmul.ac.uk/Atlas

[86] R.A. Wilson. Computing in the Monster. In *Groups, Combinatorics & Geometry (Durham, 2001)*, 327–335, World Sci. Publishing, River Edge, NJ, 2003.

[87] K. Zsigmondy. Zur Theorie der Potenzreste, *Monatsh. für Math. u. Phys.*, 3:265–284, 1892.

E.A. O'Brien, Department of Mathematics, University of Auckland, Private Bag 92019, New Zealand

Email: obrien@math.auckland.ac.nz