The use of permutation representations in structural computations in large finite matrix groups

John J. Cannon

School of Mathematics and Statistics, University of Sydney, NSW 2006, Australia

Derek F. Holt

Mathematics Institute, University of Warwick, Coventry CV4 7AL, UK

William R. Unger

School of Mathematics and Statistics, University of Sydney, NSW 2006, Australia

Abstract

We determine the minimal degree permutation representations of all finite groups with trivial soluble radical, and describe applications to structural computations in large finite matrix groups that use the output of the `CompositionTree` algorithm. We also describe how this output can be used to help find an effective base and strong generating set for such groups. We have implemented the resulting algorithms in Magma, and we report on their performance.

Keywords: Permutation representation, finite matrix group, structural computation

1. Introduction

The main theoretical result in this paper is the determination of the smallest degree of a faithful permutation representation of any finite group that has no nontrivial soluble normal subgroups. We were motivated to investigate this question by its applications to structural computations in large finite matrix groups.

The paper [1] contains a detailed description of the `CompositionTree` algorithm for computing a *composition tree* of a finite group G. A composition tree consists (roughly) of a composition series of G together with some associated data. The algorithm is principally intended for the investigation of large finite matrix groups $G \leq \mathrm{GL}(d, q)$, but it can also be applied to finite permutation groups. The successful computation of a composition tree for G enables *constructive membership testing* (that is, testing elements $g \in \mathrm{GL}(d, q)$ for membership of G and writing $g \in G$ as words over a specified generating set X of G), identifying the composition factors of G as abstract simple

Email addresses: `j.cannon@sydney.edu.au` (John J. Cannon), `D.F.Holt@warwick.ac.uk` (Derek F. Holt), `william.unger@sydney.edu.au` (William R. Unger)

The authors acknowledge the support of Australian Research Council grant DP130104534

groups, and setting up constructive isomorphisms between the nonabelian composition factors and their standard copies.

The paper [1] also contains descriptions of some applications of the output of `CompositionTree`, including the computation of a chief series, the soluble radical, and Sylow subgroups of G. We shall refer to structural computations in a matrix group G that use this output as *`CT-SR` algorithms*, since they all make use of a certain chief series for G that passes through the soluble radical L of G. Many of these algorithms can be rendered significantly more effective if a permutation representation of G/L of reasonably small degree can be found, and we shall show how our theoretical result mentioned above can be used in practice to compute effectively a permutation representation of G/L of smallest possible degree.

Prior to the availability of `CompositionTree`, almost all structural computations in finite matrix groups were carried out, by default, using base and strong generator set (BSGS) data structures, which were introduced originally by Sims [26] for computing in finite permutation groups. These have been very effective in groups for which a BSGS with reasonably short basic orbits is readily available. Some matrix groups, such as $\mathrm{GL}(d, q)$ for even moderately large d and q, have no such BSGS. But there are many examples of large matrix groups in which a good BSGS exists but is difficult to find. The other main topic of this paper is the description of a `CT-SR` algorithm that uses the `CompositionTree` data together with a permutation representation of G/L to help in the search for a good BSGS. This has turned out to be highly effective in a number of groups, particularly in examples where both L and G/L are moderately large.

We also include a brief description of a number of new `CT-SR` algorithms, including the computation of conjugacy classes, character tables, and subgroups of bounded index. These are essentially routine applications of existing methods for use with the `CompositionTree` data (although their implementation was often time-consuming).

`CompositionTree` has been implemented in Magma [3], as have our applications, which are all available in releases V2.23 upwards.

A more theoretical treatment of computation in matrix groups over finite fields, which also involves computing in the radical quotient G/L, is presented in [2]. A related but slightly different approach to practical computation is described in [23], and the algorithms presented there have been implemented in GAP [14] in the package `recog` [24]. Some of the methods that we describe in this paper, including the use of a permutation representation of G/L for the computation of conjugacy classes of G, have been implemented by Hulpke in `recog`, and are presented in [17].

The restriction of these methods to matrix groups over *finite* fields is not serious, because the algorithms described in [12] allow us easily to map a finite subgroup of $\mathrm{GL}(d, K)$ for an infinite field K to an isomorphic subgroup of $\mathrm{GL}(d, q)$ for a suitable q.

We prove our results about minimal degree permutation representations of groups with trivial soluble radical in Section 2. Our `CT-SR` algorithm that attempts to find a useful BSGS is described in Section 3.

The new `CT-SR` algorithms are described briefly in Section 4. Although they are based on existing methods, we note that we are now able to routinely compute character tables of groups that have been, until now, outside of the range of default general purpose methods, such as the subgroup $G = 2^{1+22}.\mathrm{Co}_2$ of the "Baby Monster" sporadic simple group, with G given as a subgroup of $\mathrm{GL}(1025, 2)$. Our method for finding subgroups of bounded index includes a new feature (which is also useful in calculations with permutation groups) that enables us to avoid unnecessary computations in simple composition factors that do not themselves have any proper subgroups up to that index.

Finally, in Section 5, we report on the performance of our Magma implementations.

2. Computing a permutation representation of the radical quotient G/L

2.1. Calculating the minimal permutation degree: theory

For a finite group G, we define $m(G)$ to be the smallest degree of a faithful permutation representation of G. Particularly when G is insoluble, a low degree permutation representation, if it exists, is likely to be the most useful representation for carrying out effective structural computations in G, and so calculating (or at least estimating) $m(G)$ is important for this purpose. Our first aim in this section is to determine $m(G)$ for all finite groups G with trivial soluble radical. For general finite groups G this problem appears to be significantly more difficult.

For a subgroup $H \leq G$, we recall that $\mathrm{Core}_G(H)$ is the smallest normal subgroup of G that is contained in H, and H is said to be *core-free* in G if $\mathrm{Core}_G(H) = 1$. If G is simple or, more generally, if G has a unique minimal normal subgroup, then a minimal degree faithful permutation representation of G is transitive, and $m(G)$ is equal to the smallest index in G of a core-free subgroup. In general, a minimal degree representation may be intransitive.

We have $m(\mathrm{A}_n) = n$ for $n \geq 5$, and $m(G)$ is known for all sporadic finite simple groups; see [10]. Based on the study of the maximal subgroups of the classical simple groups and of the exceptional simple groups of Lie type in [18] (based on results in [11] and [25]) and [19], respectively, $m(G)$ has been determined for simple groups in these classes in a series of papers by Mazurov and Vasilyev [20, 21, 28, 29, 30]. The results are presented in [15, Table 4]. So $m(S)$ is known for all finite simple groups S, and we observe that $m(S) \geq 5$ and $m(S)^2 < |S|$ for all such S.

Lemma 2.1. *Let $G = \times_{i=1}^k S_i$ be a direct product of finite nonabelian simple groups S_i, and let $m_i = m(S_i)$. Then:*

(i) The smallest index of a core-free subgroup of G is $\prod_{i=1}^k m_i$.
(ii) $m(G) = \sum_{i=1}^k m_i$.

Proof. (i) We use induction on k. The result is clear for $k = 1$, so suppose that $k > 1$. Each S_i has a subgroup H_i of index m_i, and $\times_{i=1}^k H_i$ is a core-free subgroup of G of index $\prod_{i=1}^k m_i$. So we need to prove that any core-free subgroup H of G has index at least $\prod_{i=1}^k m_i$.

If the projection of H onto each S_i is a proper subgroup of S_i, then this projection has index at least m_i in S_i, and the result follows. Otherwise, H projects onto at least one of the S_i, which we can assume to be S_1. Then $H \cap S_1 \trianglelefteq S_1$, and H core-free implies that $H \cap S_1 \neq S_1$, so $H \cap S_1 = 1$. Let ρ and ρ' be the projections of H onto S_1 and $S_1' := \times_{i=2}^k S_i$, respectively. So $\ker(\rho') = H \cap S_1 = 1$.

If $\mathrm{Im}(\rho')$ is core-free in S_1' then, by induction, $\mathrm{Im}(\rho')$ has index at least $\prod_{i=2}^k m_i$ in S_1' and index at least $|S_1| \prod_{i=2}^k m_i > \prod_{i=1}^k m_i$ in G, and the result follows, since $H \cong \mathrm{Im}(\rho')$.

Otherwise, $C := \mathrm{Core}_{S_1'}(\mathrm{Im}(\rho'))$ is a nontrivial direct product of some of the S_i. But, since H is core-free in G, the kernel of ρ restricted to $\rho'^{-1}(C)$ is trivial. But, since $\rho'^{-1}(C) \trianglelefteq H$, we have $\rho(\rho'^{-1}(C)) \trianglelefteq \rho(H) = S_1$, so $\rho(\rho'^{-1}(C)) = S_1$. Hence C must be equal to one of the S_i, say $C = S_2$, with $S_2 \cong S_1$, and $\rho'^{-1}(C)$ is a diagonal subgroup of $S_1 \times S_2$. Now this subgroup is maximal and self-normalising in $S_1 \times S_2$, and so it must be equal to the projection of H onto $S_1 \times S_2$.

Since $m_1^2 < |S_1|$, the result now follows if $k = 2$. Otherwise, $\mathrm{Im}(\rho')/C = \mathrm{Im}(\rho')/S_2$ is a core-free subgroup of $S_1'/S_2 \cong \times_{i=3}^k S_i$ (or else C would be larger) and, by induction, $\mathrm{Im}(\rho')/S_2$ has index at least $\prod_{i=3}^k m_i$ in S_1'/S_2. So $\mathrm{Im}(\rho')$ has index at least $|S_1| \prod_{i=3}^k m_i \leq \prod_{i=1}^k m_i$ in G, which completes the proof.

(ii) This is also proved in [13, Theorem 3.1]. Observe first that G has a faithful permutation representation of degree $\sum_{i=1}^k m_i$ with k orbits, and with S_i acting faithfully of degree m_i on the i-th orbit and trivially on all other orbits. So we need to prove that, if G acts faithfully on a set Ω, then $|\Omega| \geq \sum_{i=1}^k m_i$. We prove this by induction on k. The case $k = 1$ is clear, so assume that $k > 1$.

Since $\sum_{i=1}^k m_i < \prod_{i=1}^k m_i$, the result follows from (i) if the action is transitive. Otherwise $\Omega = \Omega_1 \cup \Omega_2$ where the subsets Ω_1 and Ω_2 are both fixed by G. Since each S_i must act faithfully on at least one of Ω_1 and Ω_2, the result follows by induction applied to these two actions. □

Recall that a group A is *almost simple* if $S \leq A \leq \mathrm{Aut}(S)$ for some finite nonabelian simple group S. The proof of the following result consists of a similar analysis as for the simple groups, and will be omitted.

Proposition 2.2. *Let $S \leq A \leq \mathrm{Aut}(S)$ for a finite nonabelian simple group S. Then $m(A) = m(S)$, except in the following cases*

S	A	$m(S)$	$m(A)$
A_6	$A \not\leq \mathrm{S}_6$	6	10
$\mathrm{PSL}(2,7)$	$\mathrm{PGL}(2,7)$	7	8
M_{12}	$\mathrm{Aut}(\mathrm{M}_{12}) = \mathrm{M}_{12}.2$	12	$2m(S)$
O'N	$\mathrm{Aut}(\mathrm{O'N}) = \mathrm{O'N}.2$	122760	$2m(S)$
$\mathrm{PSL}(d,q)$, $d \geq 3$			
$(d,q) \neq (3,2),(4,2)$	$A \not\leq \mathrm{P\Gamma L}(d,q)$	$\frac{q^d-1}{q-1}$	$2m(S)$
$\mathrm{PSp}(4,2^e)$, $e \geq 2$	$A \not\leq \mathrm{P\Gamma Sp}(4,2^e)$	$\frac{2^{4e}-1}{2^e-1}$	$2m(S)$
$\mathrm{PSU}(3,5)$	$A \not\leq \mathrm{P\Sigma U}(3,5)$	50	126
$\mathrm{P\Omega}^+(2d,3)$, $d \geq 4$	$3 \not\mid \lvert A/S\rvert$, $(*)$	$\frac{3^{d-1}(3^d-1)}{2}$	$\frac{(3^d-1)(3^{d-1}+1)}{2}$
$\mathrm{P\Omega}^+(8,q)$, $q \geq 4$	$3 \mid \lvert A/S\rvert$	$\frac{(q^4-1)(q^3+1)}{q-1}$	$3m(S)$
$\mathrm{P\Omega}^+(8,2)$	$3 \mid \lvert A/S\rvert$	120	$3m(S)$
$\mathrm{P\Omega}^+(8,3)$	$3 \mid \lvert A/S\rvert$, $6 \not\mid \lvert A/S\rvert$	1080	$3m(S)$
$\mathrm{P\Omega}^+(8,3)$	$6 \mid \lvert A/S\rvert$	1080	3360
$\mathrm{G}_2(3)$	$\mathrm{Aut}(\mathrm{G}_2(3))$	351	$2m(S)$
$\mathrm{G}_2(3^e), e > 1$	$A \not\leq \Gamma\mathrm{G}_2(3^e)$	$\frac{q^6-1}{q-1}$	$2m(S)$
$\mathrm{F}_4(2^e)$	$A \not\leq \Gamma\mathrm{F}_4(2^e)$	$\frac{(q^{12}-1)(q^4+1)}{q-1}$	$2m(S)$
$\mathrm{E}_6(q)$	$A \not\leq \Gamma\mathrm{E}_6(q)$	$\frac{(q^9-1)(q^8+q^4+1)}{q-1}$	$2m(S)$

The Condition $(*)$ in the table is $A \not\leq \mathrm{PGO}(2d,3)$ when $d > 4$, and A is not contained in any conjugate of $\mathrm{PGO}(8,3)$ in $\mathrm{Aut}(\mathrm{P\Omega}^+(8,3))$ when $d = 4$.

In particular, we have $m(A) < 4m(S)$ in all cases.

Remark. In fact we have $m(A) \leq 28m(S)/9$ in all cases, with equality for certain subgroups of $\mathrm{Aut}(\mathrm{P\Omega}^+(8,3))$.

The following result can be proved by a routine examination of the examples in the table.

Lemma 2.3. *For $1 \leq i \leq k$ with $k > 1$, let S_i be a finite nonabelian simple group, and let A_i be an almost simple group with $S_i \leq A_i \leq \mathrm{Aut}(S_i)$. Then $\prod_{i=1}^k m(S_i) > \sum_{i=1}^k m(A_i)$.*

Next we consider the case when the group G is not almost simple, but has a unique minimal normal subgroup M, which is insoluble. So $M = \times_{i=1}^k S_i$, where the S_i are isomorphic nonabelian simple

groups that are permuted transitively under the conjugation action of G. In [7, Section 3.2], it is shown that there is an embedding $\psi : G \to W := \mathrm{Aut}(S_1) \wr H$ that maps M isomorphically onto the socle of W, where H is the transitive subgroup of S_k induced by the conjugation action of G on the set $\{S_i : 1 \leq i \leq k\}$. By looking at the definition of ψ, we see that $\mathrm{Im}(\psi) \leq A_1 \wr H$, where $S_1 \leq A_1 \leq \mathrm{Aut}(S_1)$, and A_1 is the subgroup of $\mathrm{Aut}(S_1)$ induced by the conjugation action of $N_G(S_1)$ on S_1.

Lemma 2.4. *Let G, M, S_i, H, A_1 be as above. Then $m(G) = km(A_1)$.*

Proof. The natural permutation representation of $A_1 \wr H$ arising from a representation of A_1 of degree $m(A_1)$ is faithful of degree $km(A_1)$, so we just need to prove that $m(G) \geq km(A_1)$.

So suppose that we have a faithful permutation representation of G on a set Ω, and identify G with its image in $\mathrm{Sym}(\Omega)$. Since G has a unique minimal normal subgroup, a minimal degree faithful permutation representation of G is transitive, so we may assume that G is transitive. Then G acts transitively on the orbits of M on Ω, and so there is a fixed number j such that precisely j of the S_i act nontrivially on each such orbit. Since each S_i acts nontrivially on some orbit of M, there must be at least k/j orbits in total. By Lemma 2.1 (i), each such orbit Δ satisfies $|\Delta| \geq m(S_1)^j$ and so $|\Omega| \geq km(S_1)^j/j$ which, by Lemma 2.3, is greater than $km(A_1)$ when $j > 1$. So we may assume that $j = 1$. In other words, the subgroups S_i have mutually disjoint supports.

Now let $N_1 = N_G(S_1)$, and let Ω_1 be an orbit of N_1 on which S_1 acts nontrivially. Then S_1 acts nontrivially on each of its orbits in Ω_1, so all S_i with $i > 1$ act trivially on Ω_1. For each i with $1 \leq i \leq k$, we define $\Omega_i = \Omega_1^{g_i}$, for $g_i \in G$ with $S_1^{g_i} = S_i$ (which is independent of the choice of g_i). Since the Ω_i are disjoint, we have $|\Omega| \geq k|\Omega_1|$.

Suppose that S_1 has exactly r orbits on Ω_1 (so M has kr orbits on Ω). If $r > 3$ then, by Lemma 2.1 (ii) and Proposition 2.2, $|\Omega| \geq 4km(S_1) > km(A_1)$ as required, so assume that $r \leq 3$. Also, if S_1 acts imprimitively on these orbits, with blocks of size $b > 1$, then the orbits have length at least $bm(S_1)$, so we can assume that $br \leq 3$ and hence $r = 1$ in that case. Let $\bar{N}_1$ and $\bar{C}_1$ be the induced actions of N_1 and $C_1 := C_G(S_1)$ on Ω_1. Since the centraliser of S_1 in its action on each of its orbits has order at most $b!$, we have $|\bar{C}_1| \leq 6$, and S_1 must permute the orbits of $\bar{C}_1$ nontrivially. Then $\bar{N}_1$ induces a faithful action of $\bar{N}_1/\bar{C}_1 \cong A_1$ on the orbits of $\bar{C}_1$, so there are at least $m(A_1)$ such orbits, and hence $|\Omega_1| \geq m(A_1)$, which completes the proof. □

Now let G be an arbitrary finite group with trivial soluble radical, and let $M = M_1 \times \cdots \times M_r$ be the socle of G, where $M_1, \ldots, M_r$ are the minimal normal subgroups of G. Then each M_i is the direct product of isomorphic simple groups $S_{i1}, \ldots, S_{ik_i}$ for some $k_i \geq 1$, which are permuted transitively under the conjugation action of G. Furthermore, G is the subdirect product of groups $G_1, \ldots, G_r$, where G_i has the unique minimal normal subgroup M_i. For each i, let A_{i1} be the group with $S_{i1} \leq A_{i1} \leq \mathrm{Aut}(S_{i1})$ that is induced by conjugation in $N_G(S_{i1})$.

Proposition 2.5. *With the notation in the previous paragraph, we have $m(G) = \sum_{i=1}^r k_i m(A_i)$.*

Proof. Since, by Lemma 2.4, G is a subdirect product of groups G_i with $m(G_i) = k_i m(A_i)$, we have $m(G) \leq \sum_{i=1}^r k_i m(A_i)$. So suppose that we have a faithful permutation representation of G on a set Ω, and identify G with its image in $\mathrm{Sym}(\Omega)$. We have to prove that $|\Omega| \geq \sum_{i=1}^r k_i m(A_i)$. The proof is by induction on r, and Lemma 2.4 handles the case $r = 1$, so we suppose that $r > 1$.

Suppose first that, for all orbits Δ of G, the M_i do not all act faithfully on Δ. Let Δ be an orbit of G. By renumbering, we may assume that $M_1, \ldots, M_s$ act faithfully on Δ, and that $M_{s+1}, \ldots, M_r$ do not. Since they are minimal normal subgroups, $M_{s+1}, \ldots, M_r$ must act trivially on Δ. We

claim that $|\Delta| \geq \sum_{i=1}^{s} k_i m(A_i)$. Since each M_i must act nontrivially on some orbit of G, this is enough to prove the result.

Let $M' = M_1 \times \cdots \times M_s$. Using similar arguments to the proof of Lemma 2.4, if M' has more than three orbits on Δ, then the claim follows from Lemma 2.1 (ii) and Proposition 2.2. Then, putting $C = C_G(M')$ and $\bar{C}$ the induced action of C on Δ, we have $|\bar{C}| \leq 6$, and M' must faithfully permute the orbits of $\bar{C}$. So the induced action of G on these orbits is a subdirect product of the groups $G_1, \ldots, G_s$ and, since we are assuming that $s < r$, the claim follows from the inductive hypothesis.

Otherwise, there is an orbit Δ of G on which each M_i acts faithfully. Let Δ_M be an orbit of M with $\Delta_M \subseteq \Delta$. For $1 \leq i \leq k$, suppose that precisely j_i of the factors $S_{i1}, \ldots S_{ik_i}$ of M_i act faithfully on Δ_M. Then $j_i \geq 1$ for all i.

By Lemma 2.1 (i), we have $|\Delta_M| \geq \prod_{i=1}^{k} m(S_{i1})^{j_i}$ which, by Lemma 2.3, is greater than $\sum_{i=1}^{k} j_i m(A_{i1})$. Now, since each S_{ij} must act nontrivially on Δ_M^g for some $g \in G$, the total number of orbits of M of the form Δ_M^g must be at least k_i/j_i for each individual i. So $|\Omega| \geq |\Delta| > \sum_{i=1}^{k} k_i m(A_{i1})$, which completes the proof. □

2.2. Calculating the minimal degree permutation representation: practice

The nonabelian composition factors of G are identified as abstract simple groups by `CompositionTree`. Each nonabelian simple group S has a designated *standard copy*, which is (the image of) either a permutation representation of S, or a projective matrix representation of S over a finite field. For example, the standard copy of A_n is its natural permutation representation, and the standard copy of $\mathrm{PSL}(d, q)$ is $\mathrm{SL}(d, q)/Z(\mathrm{SL}(d, q))$, with the natural representation of $\mathrm{SL}(d, q)$. `CompositionTree` sets up effective isomorphisms between the nonabelian composition factors of G and their standard copies: that is, isomorphisms for which images and inverse images of arbitrary group elements can be efficiently computed.

Let L be the soluble radical of G. The structure of G/L as a subdirect product of groups with unique minimal normal subgroups is computed using the algorithms described in [1, Sections 10,11], and we use the notation of the previous subsection for the minimal normal subgroups M_i of G/L, and their simple factors S_{ij}. In particular, we can identify the automorphisms of the groups S_{i1} induced by elements of $N_{i1} := N_G(S_{i1})$, and hence determine the subgroups A_{i1} of $\mathrm{Aut}(S_{i1})$ that are induced by N_{i1}.

The results established in the previous subsection enable us to calculate $m(G/L) = \sum_{i=1}^{r} k_i m(A_i)$. By the methods described in [7, Section 3.2] in the context of permutation groups, and also in [17] in the context of matrix groups with a composition tree, we can now compute $\rho : G \to \mathrm{Sym}(\Omega)$ with $\ker \rho = L$ that induces a minimal degree permutation representation of G/L provided that, for each i with $1 \leq i \leq r$, we can

(i) construct a minimal degree permutation group $A_{i1}^{\mathsf{p}} \cong A_{i1}$; and
(ii) compute images under a surjective homomorphism $\rho_i : N_{i1} \to A_{i1}^{\mathsf{p}}$.

Since we shall be discussing a single value of i from now on, let us drop the first subscript, and write S_1 instead of $S_{i1},$, etc. We shall now discuss how we accomplish (i) and (ii) for the various types of simple groups S_i. We denote by S_1^{s} the standard copy of S_1 computed by `CompositionTree`. So we have an effective isomorphism $X \to S_1^{\mathsf{s}}$ with kernel Y, where X/Y is the composition factor of G corresponding to S_1.

2.2.1. Alternating groups

Both tasks are straightforward for $S_1 \cong \mathrm{A}_n$. Apart possibly from the small cases $n = 5, 6, 8$, where A_n has an alternative name, we have $S_1^{\mathsf{s}} = \mathrm{Alt}(n)$ (the natural representation of A_n), and we take $A_1^{\mathsf{p}} = \mathrm{Alt}(n)$ or $\mathrm{Sym}(n)$ in the natural representation. The image of an automorphism in $\mathrm{Sym}(n)$ of S_1^{s} can be found easily by evaluating it on 3-cycles of $\mathrm{Alt}(n)$.

2.2.2. Classical groups

Suppose next that S_1 is isomorphic to a classical simple group, and let $\hat{S}_1$ be the corresponding quasisimple matrix group. (So, for example, if $S_1 \cong \mathrm{PSL}(d, q)$, then $\hat{S}_1 = \mathrm{SL}(d, q)$ in its natural representation.) Then (except possibly for a few very small cases) $S_1^{\mathsf{s}} = \hat{S}_1/Z(\hat{S}_1)$.

Let V be the vector space on which $\hat{S}_1$ acts naturally. Then, with the exception of a few cases such as $\mathrm{PSU}(3, 5)$ and $\mathrm{PSp}(2m, 2)$ for $m \geq 3$, the restriction of the smallest degree permutation representation of A_1 to S_1 corresponds to the action of $\hat{S}_1$ on the subspaces of V spanned by singular or non-singular vectors of V or, when $\hat{S}_1 = \mathrm{SU}(4, q)$, the singular 2-subspaces of V. We can use this action to define an effective homomorphism with kernel $Z(\hat{S}_1)$ from $\hat{S}_1$ to a permutation group $P \cong S_1$, which provides an effective isomorphism $S_1^{\mathsf{s}} \to P$. (In our current implementations, we do not attempt to achieve a representation of smallest possible degree in the small exceptional cases.)

Any diagonal or field automorphisms of S_1 in A_1 induce (linear or semilinear) actions on V, and so their actions on the subspaces can also be computed easily, and we can extend P to a group Q of the same degree containing these automorphisms. (There are technical complications in the groups $\Omega^-(d, q)$ resulting from the fact that field automorphisms do not always preserve the matrix of the fixed orthogonal form, but we will not discuss those further here.)

If there are no graph automorphisms in A_1, then $\deg Q = m(A_1)$, and we can take $A_1^{\mathsf{p}} = Q$, thereby solving Problem (i). Otherwise, we have $m(A_1) = 2 \deg Q$ or, in the case of the triality automorphism of $\Omega^+(8, q)$, $m(A_1) = 3 \deg Q$. The actions of the graph automorphisms on S_1^{s} in the various cases are described precisely in [9, Section 12], and we can construct A_1^{p} as a subgroup of $Q \wr C_2$ or $Q \wr \mathrm{Sym}(3)$, using essentially the same wreath product embedding techniques as described earlier.

For the construction of the homomorphisms ρ_i in Problem (ii), we observe first that the `CT-SR` data enables us to compute the actions of the automorphisms of S_1 induced by elements of N_1 as automorphisms of S_1^{s}. We can use the methods described in [1, Section 10] to express these automorphisms as products of graph, field, diagonal and inner automorphisms. Our construction of A_1^{p} allows us to map (under ρ_i) the first three of these to the corresponding automorphisms in A_1^{p}. The inner automorphism in the product is returned as an element of $\hat{S}_1$ that induces it, and we can either use the `CT-SR` data to express it as a word (or more precisely a straight line program) in the generators of $\hat{S}_1$ and thereby compute its image under ρ_i, or (in almost all cases) we can compute this image directly by its action on the relevant subspaces of V.

2.2.3. Exceptional groups of Lie type and sporadic groups

Similar methods to those for the classical groups could be used for the exceptional groups of Lie types and for the Suzuki groups. We have not yet implemented these (although it would be worthwhile to do this at least for the Suzuki groups, which should not be difficult), and we currently treat those groups in these classes that fall within the range of practical computation as sporadic groups.

There is enough information in [10] and [31] to solve Problem (i) and to find an effective isomor-

phism from S_1^{s} to its isomorphic image in A_1^{p}. Indeed, this isomorphism is typically equal to the identity map when S_1^{s} is a permutation group and to an action on subspaces of a vector space when S_1^{s} is a matrix group.

For Problem (ii), we have to identify an automorphism α of S_1^{s}, and we currently do this by calculating the action of α on the two *standard generators* of S_1^{s}, mapping their images to A_1^{p} and then identifying the element of A_1^{p} which induces this automorphism by carrying out two conjugacy tests in A_1^{p}. These conjugacy tests are unsatisfactorily slow in some large groups A_1^{p} (such as O′N.2, which has minimal degree 245520), and improved methods would be desirable.

2.3. Concluding remarks

For groups A_1 for which $m(A_1)$ is large, evaluation of the maps ρ_i, and hence also of the complete permutation action map $\rho : G \to \mathrm{Sym}(\Omega)$ can be slow, particularly in the large sporadic groups. So we need to design subsequent algorithms to use as few applications of ρ as possible. This was also observed by Hulpke in [17], who recommended using the *shadowing* technique wherever possible, which essentially means that, if we need to evaluate $\rho(g_1 g_2 \cdots g_n)$ and we have already computed each $\rho(g_i)$, then we should store their values and compute it as $\rho(g_1)\rho(g_2)\cdots\rho(g_n)$.

In contrast to this, an inverse image under ρ can typically be evaluated much more quickly, by using BSGS techniques in A_1^{p} to express elements of A_1^{p} as words over a strong generating set. Also, if we successfully use the data computed so far to find a satisfactory base for G using the methods to be described in the following section, then we can redefine the map ρ using BSGS machinery, and its subsequent evaluation will be significantly faster.

3. Using `CT-SR` algorithms to compute a BSGS

The purpose of this section is to describe some new methods, that use `CT-SR` data, to find an effective BSGS of a matrix group over a finite field. We shall do that in Subsection 3.1 below, but let us first briefly survey some earlier methods.

Let $G \leq \mathrm{GL}(d, K)$ be a finite matrix group over a field K, and let $V \cong K^d$ be the space of row vectors on which G acts (on the right). A *base* for V consists of a sequence $(\beta_1, \ldots, \beta_k)$ of subspaces and vectors from V such that only the identity element stabilises every β_i. Let $G^{(i)} := G_{\beta_1,\ldots,\beta_{i-1}}$ be the i-th basic stabiliser; so $G^{(k+1)} = 1$.

For efficient computation within G, the *basic orbits* $\beta_i^{G^{(i)}}$ should be moderately short (up to about 10^4 is desirable, but computation remains feasible with basic orbit lengths exceeding 10^6). Another consideration is that computation is more efficient when the dimensions of the subspaces β_i are not too large, and dimension 1 is the optimal choice. This is mainly because, when working with bases, large numbers of images of base points under group elements need to be tested for equality, and testing equality of subspaces requires the computation of echelonised bases. So, in practice, there is frequently a choice to be made between larger dimensional β_i and longer basic orbit lengths. Our (admittedly limited) experience to date suggests that it is preferable to choose lower dimensional β_i provided that this does not result in excessively large basic orbit lengths.

Let us call a base for G *satisfactory* if it can be used for effective computations within G without being excessively slow or memory intensive. This definition is necessarily imprecise, because the effectiveness of a base depends heavily on what types of computations in the group it will be used for. Not all finite matrix groups of moderately small dimension have satisfactory bases. For example, for $G = \mathrm{SL}(d, q)$, there will inevitably be a basic orbit of length at least $(q^d - 1)/(q - 1)$,

and at least d base points with orbit lengths of this order of magnitude. But there are many matrix groups that have satisfactory bases that are not straightforward to find, and in this section we shall describe some techniques for locating them.

The first implementations of BSGS methods for computing with a matrix group G over a finite field were described by Butler in [5]. The base computed is a combination of 1-dimensional subspaces and vectors from V. These subspaces were typically chosen as those spanned by basis vectors of V. Methods for finding subspaces with shorter orbits, by choosing them as common eigenspaces of two or more group elements, were proposed by Murray and O'Brien in [22] and their implementations are available in Magma.

More recently, methods involving the algorithms used in `CompositionTree` have been used in the case when the field K is finite. The idea is to first apply a suitably chosen basis change to V and then to choose (some of) the β_i as the 1-dimensional subspaces spanned by basis vectors.

If the action of G on V is reducible, then it is helpful (in the sense that the resulting basic orbit lengths will be shorter) to choose a basis for V that is compatible with a composition series for the action of G, and to choose the β_i from small dimensional G-submodules. Again, if the action of G is imprimitive, then it helps to choose a basis for V that contains bases of the blocks of imprimitivity.

In the current Magma implementation of the `CT-SR` algorithms, before embarking on a computation with a new group, Magma will, by default, use these methods to try and find a satisfactory base for the group. If it succeeds, then it will use that base for computations, and otherwise it will use `CT-SR` algorithms. Unfortunately, the best choice may depend on the nature of the subsequent computations with that group, which is obviously impossible for Magma to predict, so the user always has the option of specifying which methods to use.

3.1. Using the `CT-SR` data to search for a base

Even more recently, the `CT-SR` data have been used to help search for a satisfactory base for matrix groups over a finite field. As we shall see, this new method involves making a variety of choices at all stages, and further research will be needed to develop guidelines for making effective choices. Unlike any of the methods described above, the chosen base may contain subspaces β_i of dimension greater than 1. So it is sometimes necessary to make a choice between higher dimensional β_i and longer basic orbit lengths.

The new method has been successful in finding usable bases for almost all of the examples that are discussed in Section 5 below, so we also have a choice between using BSGS methods and the `CT-SR` data for the algorithms to be described in Section 4. The algorithm descriptions do not depend on which of these choices is made, but their implementations might. Our experience to date with Magma implementations is that the examples typically run faster using BSGS techniques but not by a factor greater than 4. We observe also that a considerably larger variety of algorithms is available in the BSGS context: for example we can compute conjugacy class representatives of all subgroups rather than just subgroups of a specified type as in Section 4.

The main idea of this new method is to find subgroups $H < G$ of low index $|G : H|$, and look for subspaces W of V that are left invariant by H but not by G. If we find such a subspace W, then the orbit length of W will be at most $|G : H|$ so W may be a suitable choice for a base point for G. (But, for reasons mentioned earlier, if $\dim(W)$ is high, then we may prefer to defer this choice and first consider subgroups of larger index that might produce lower dimensional base points.) Provided that we consider subgroups H in order of decreasing size, we will know that H is equal to the full stabiliser in G of W. We can then either apply the methods recursively to H or alternatively one of the earlier methods for choosing a BSGS may be effective on H.

We start by running `CompositionTree` on the group and then using the `CT-SR` data to find the homomorphism onto the radical quotient G/L and also a permutation representation of G/L as described in Section 2.

In the case when G/L is trivial (that is, when G is soluble), the `CT-SR` data includes a representation of G as a PC-group, and we can use this effectively to find the required subgroups of G of low index. We remark however that the earlier methods for finding a BSGS, and specifically those that involve a basis change of V to respect a decomposition of G as a reducible or an imprimitive group, are typically highly effective in this situation.

Otherwise G/L is nontrivial and we can use the low index subgroups algorithm to find subgroups of low index in the permutation representation of G/L, and then use their inverse images in G as our candidate subgroups H. In most of the examples that we have considered, this approach has been successful in finding a satisfactory base for G. In the small number of examples in which it failed (this happened for example in the example $2^{14}.\mathrm{A}_6^3.\mathrm{S}_3 < \mathrm{GL}(64,5)$ in the tables in Section 5), we were able to apply the method recursively to a subgroup H of low index in G for which the soluble radical of H is larger than that of G, and it succeeded on H.

4. Some further `CT-SR` algorithms

It is assumed throughout this section that a permutation representation ρ of G/L has been computed as described in Section 2. The algorithms discussed can be carried out either using the `CT-SR` data, or with BSGS methods if a satisfactory base for G is available.

4.1. Centralisers, conjugacy testing, normalisers and intersections

We compute centralisers in G of elements of G, test pairs of elements of G for conjugacy in G, and find a set of representatives of the conjugacy classes of G by solving these problems in the permutation group $\mathrm{Im}(\rho)$ and then lifting the results through elementary abelian layers of L, using the algorithms described in [16, Section 8.8] for polycyclic groups. These methods are also described for matrix groups in [17, Section 6]. Algorithms for computing normalisers in G of subgroups of G, testing subgroups of G for conjugacy in G, and computing the intersection of two subgroups of G, were implemented using the same general approach.

4.2. Character tables

An algorithm developed by Unger [27] for computing the table of complex characters for a group G is based on Brauer's result that every character of a finite group G can be written as an integer linear combination of characters induced from elementary subgroups of G. An *elementary subgroup* is one that is a direct product of a cyclic group and a group of prime-power order.

At the time the algorithm was published it was seen as being applicable only to groups of moderate size. However, a deeper understanding of the algorithm and a number of improvements have led to a more powerful algorithm that is capable of computing character tables of very large groups provided that the number of conjugacy classes does not much exceed three thousand.

Recently attention has been drawn to the fact that, while numerous theorems have been proven using information taken from information given in the ATLAS of Finite Groups [10], there are many cases in which no proofs are given for the correctness of this information, and no citations are provided. This general problem and its recent and potential remedies are discussed in [4]. In particular, independent computations of the ATLAS character tables are being carried out,

including their verification using the Unger algorithm. More than 400 of the approximately 430 ATLAS character tables have now been verified. These include all but two of the sporadic groups. A mixture of permutation group BSGS methods and matrix group `CT-SR` methods are being used. The only serious error found so far was in the case of the character table for $E_6(2)$ where the 2-power map of six conjugacy classes of elements of order 91 had been incorrectly determined leading to errors in six characters. A paper describing this work is in preparation.

4.3. Normal subgroups, maximal subgroups, coset actions, and low index subgroups

To compute maximal subgroups and normal subgroups of matrix groups for which the `CT-SR` data has been computed, we use the same methods as are described for permutation groups in [7] and [6], respectively. The same applies to the computation of the permutation action $\tau_{G,H}$ of the group G on the right cosets of a subgroup H, which we can effectively split up into the corresponding computations for the subgroup HL/L of GL/L and for the subgroup $H \cap L$ of L, in their representations as a permutation group and as a PC-group, respectively.

An algorithm for computing representatives of the conjugacy classes of subgroups of finite permutation groups up to a specified index n is described in [8]. This makes essential use of corresponding computations in G/L, and once again we can apply the same methods in the new situation. We have however introduced a new trick, which can also be used in the algorithm for permutation groups. The `CompositionTree` data includes an identification of all of the nonabelian composition factors of G and, as we saw in Section 2 the smallest degrees of their nontrivial permutation representations are all known. If, for some composition factor, this minimal degree exceeds n, then we can effectively ignore it in our search for subgroups of index up to n, since any such subgroup would have to contain that factor. As a trivial example, to find subgroups of index up to 10 in $A_5 \times A_{11}$, we can ignore the A_{11} factor and do all of our calculations in A_5. But, as is often the case, this idea is much easier to describe theoretically than it is to implement.

One motivation for the study of this problem in matrix groups is that a complete list of subgroups of index up to the degree d of an irreducible matrix group $G \leq \mathrm{GL}(d, q)$ can be used to provide a straightforward definitive test for the primitivity of G. Particularly in examples in which G is semilinear, this question is not always resolved by `CompositionTree`, and this application has already proved useful even in small matrix groups. It is often the case in such examples that some of the composition factors of G have no proper subgroups of index at most d.

5. Performance

The tables list timings of various computations using our Magma implementations of some of the functions that we have described in this paper. The computations were all done using an Intel Xeon E5-2687W CPU with a clock speed of 3.10GHz and having GB 396 of memory. All times are averages over ten runs.

The times in Table 1 are for our implementations using just the `CT-SR` data, and those in Table 2 for the same examples using the BSGS computed using this data. (So the computations whose times are listed in the columns headed 'CS' and 'RQ' in the first table were prerequisites for the computations timed in the second table.) Table 3 lists times for computing this BSGS. For the final group in the first table we were unable to find a satisfactory BSGS.

In Tables 1 and 2, the columns to the left of the vertical line provide information about the examples, whereas those to the right list times in seconds. In the structural description of the groups, the notation O for the orthogonal groups $\mathrm{O}_8^+(3)$ and $\mathrm{O}_9(3)$ denotes the simple group, which could also be denoted by $\mathrm{P}\Omega_8^+(3)$ or $\mathrm{P}\Omega_9(3)$.

G	$\lvert G\rvert$	d	q	DRQ	#Cl	CS	RQ	MS	NS	LIS	Cl	CT
$2\cdot A_{14}$	8.7E10	64	3	14	105	4.0	0.7	0.1	0.1	17	0.7	18
$O_8^+(3)$	5.0E12	298	2	1120	114	113	2.1	2.4	0.1	56	10	1422
Suz	4.5E11	142	2	1782	43	5.7	0.1	0.2	0.0	0.4	1.4	6.2
$2\cdot O_9(3)$	1.3E17	16	3	91840	326	2.3	0.1	1.4	0.0	2.9	50	189
3O'N	1.3E12	153	4	122760	80	62	48	15	4.0	35	54	211
$4.L_4(5)^3.S_3$	9.2E30	64	5	468	79547	14	14	3.3	24	237	1904	
$2^{14}.A_6^3.S_3$	4.5E12	64	5	45	1218	8.0	21	0.2	2.3	1173	37	10220
$3^{1+12}.2\cdot Suz.2$	2.9E18	78	3	1782	253	7.3	2.1	1.0	2.5	38	1034	314
$2^{9+16}.Sp_8(2)$	1.6E18	215	2	255	703	10	1.8	0.5	2.9	5.9	100	9819
$2^{30}.L_5(2)$	1.1E16	144	2	31	1033	10	0.5	0.9	2.2	54	26	16208
$2^{1+22}.Co_2$	3.5E20	1025	2	2300	448	375	50	26	117	189	4682	55616
$2^{1+11}.S_{10}(2)$	1.0E20	32	5	1023	1235	3.5	0.2	0.8	0.5	3.9	50	39443

Table 1: `CT-SR` times for subgroups, conjugacy classes and character tables

G	$\lvert G\rvert$	d	q	DRQ	#Cl	CS	MS	NS	LIS	Cl	CT
$2\cdot A_{14}$	8.7E10	64	3	14	105	4.3	0.1	0.1	0.3	0.5	6.0
$O_8^+(3)$	5.0E12	298	2	1120	114	114	1.2	0.0	0.7	12	103
Suz	4.5E11	142	2	1782	43	7.8	0.1	0.0	0.1	1.7	5.2
$2\cdot O_9(3)$	1.3E17	16	3	91840	326	3.2	1.3	0.0	5.7	43	80
3O'N	1.3E12	153	4	122760	80	69	61	25	76	46	49
$4.L_4(5)^3.S_3$	9.2E30	64	5	468	79547	34	2.0	21	13	2339	
$2^{14}.A_6^3.S_3$	4.5E12	64	5	45	1218	107	0.6	1.9	109	27	3089
$3^{1+12}.2\cdot Suz.2$	2.9E18	78	3	1782	253	9.7	2.3	4.7	45	21	182
$2^{9+16}.S_8(2)$	1.6E18	215	2	255	703	14	2.5	6.2	9.1	58	8858
$2^{30}.L_5(2)$	1.1E16	144	2	31	1033	11	2.5	17	17	55	17409
$2^{1+22}.Co_2$	3.5E20	1025	2	2300	448	606	110	180	1458	2948	51224

Table 2: BSGS times for subgroups, conjugacy classes and character tables

Notation such as 4.5E11 for a group order means 4.5×10^{11}. The columns headed 'd' and 'q' refer to the degree and field of the input group $G \leq \mathrm{GL}(d, q)$; 'DRQ' is the degree of the computed permutation representation of the radical quotient G/L; and '#Cl' is the number of conjugacy classes of G. The columns headed 'CS', 'RQ', 'MS', 'NS', 'LIS', 'Cl', and 'CT' are the times for computing a composition series (including the composition tree), the permutation representation of G/L, the maximal subgroups, the normal subgroups, the subgroups of index up to 100, the conjugacy classes, and the character table of G, respectively. We have chosen examples in which the default use of BSGS methods fails or is very slow.

In Table 3, we provide details of bases that were found for some of these groups using the method described in Subsection 3.1, after computing the `CT-SR` data and a permutation representation of the radical quotient of G. In the 'base dimensions' column, an entry 0 means that the corresponding base point was a vector rather than a subspace. Observe that most of these bases were found moderately quickly.

As a general rule our conclusion is that, if a BSGS with reasonably short orbits can be found, then it is likely to be worthwhile using it in subsequent calculations and, in difficult examples, it is worth devoting some effort to finding a good BSGS. (In a few examples, our code found bases with much shorter orbit lengths after several attempts.) But when no good base can be found, then it is better to continue using the `CT-SR` data. Indeed, there remain examples, such as $2^{1+11}.S_{10}(2)$

Group	$\lvert G\rvert$	d	q	Basic orbit lengths	Dimensions	Time
$2^{\cdot}A_{14}$	8.7E10	64	3	14 13 12 11 10 9 8 7 6 5 4 3 2	$32\ 16^{11}\ 0$	4.3
$O_8^+(3)$	5.0E12	298	2	1120 40 13 12 9 4 3^3 27^2	38 26 1^4 121 42 6 1^2	12
Suz	4.5E11	142	2	1782 416 100 63 96	36 1^4	7.9
$2^{\cdot}O_9(3)$	1.3E17	16	3	91840 120 117 108 108 81 54 2	1^7 0	3.2
3O'N	1.3E12	153	4	122760 5586 112 6 3	1^4 0	68
$4.L_4(5)^3.S_3$	9.2E30	64	5	468 155 150 125 16 312 155 150 125 156 155 150 125 16 16 4	16^{15} 0	34
$2^{14}.A_6^3.S_3$	4.5E12	64	5	288 192 96 5 5 5 4 4 4 3 3 3 4	32^{12} 0	107
$3^{13}.2^{\cdot}Suz.2$	2.9E18	78	3	22880 112 81 60 12 2 3 3 2 3^{12}	15 4^4 0 1^2 0 1^{12}	9.5
$2^{25}.S_8(2)$	1.6E18	215	2	255 126 64 32 15 8 3 2 128 128 4 32 2 2 2 2	1^{16}	5.8
$2^{30}.L_5(2)$	1.1E16	144	2	31 30 28 24 16 1024 256 2048 2	4^5 1^4	10
$2^{23}.Co_2$	3.5E20	1025	2	2300 672 165 128 72 9 2^{14} 1024	155 75 6^5 1^{14}	578

Table 3: Details of bases for G

in which we were unable to find any satisfactory BSGS using our programs.

References

[1] H. Bäärnhielm, D. F. Holt, C. R. Leedham-Green, E. A. O'Brien. A practical model for computation with matrix groups, *J. Symbolic Computation* **68** (2015), 27–60.

[2] L. Babai, R. Beals, and Á. Seress. Polynomial-time theory of matrix groups. In *STOC '09 Proceedings of the forty-first annual ACM symposium on Theory of computing*, ACM New York, 2009, 55–64.

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

[4] T. Breuer, G. Malle, and E. A. O'Brien. Reliability and reproducibility of Atlas information, 2016, `https://arxiv.org/abs/1603.08650`.

[5] G. Butler. The Schreier algorithm for matrix groups. In SYMSAC '76, *Proc. ACM Sympos. Symbolic and Algebraic Computation*, New York, 1976. (New York, 1976), Association for Computing Machinery, 167–170.

[6] J. J. Cannon, B. Cox, and D. F. Holt, Computing the subgroups of a permutation group, *J. Symbolic Comput.* **31** (2001), 149–161.

[7] J. J. Cannon and D. F. Holt. Computing maximal subgroups of finite groups, *J. Symbolic Comput.* **37** (2004), 589–609.

[8] J. J. Cannon, M. Slattery, A. K. Steel, and D. F. Holt. Computing subgroups of bounded index in a finite group, *J. Symbolic Comput.* **40** (2005), 1013–1022.

[9] R. W. Carter. *Simple Groups of Lie Type,* John Wiley and Sons, 1972.

[10] J. H. Conway, R. T. Curtis, S. P. Norton, R. A. Parker, R. A. Wilson. *ATLAS of Finite Groups*, Oxford, 1985.

[11] B. N. Cooperstein. Minimal degree for a permutation representation of a classical group, *Israel J. Math,* **30** (1978), 213–235.

[12] A. S. Detinko, D. L. Flannery, and E. A. O'Brien. Recognizing finite matrix groups over infinite fields, *J. Symbolic Comput.* **50** (2013), 100–109.

[13] D. Easdown and C. E. Praeger, On minimal faithful permutation representations of finite groups, *Bull. Austral. Math. Soc.* **38** (1988), 207–220.

[14] The GAP Group. *GAP — Groups, Algorithms, and Programming, Version 4.7*, 2014, `http://www.gap-system.org`.

[15] S. Guest, J. Morris, C. E. Praeger and P. Spiga. On the maximum orders of elements of finite almost simple groups and primitive permutation groups, *Trans. Amer. Math. Soc.* **367**, no. 11 (2015), 7665-7694.

[16] Derek F. Holt, Bettina Eick, and Eamonn A. O'Brien. *Handbook of computational group theory*, Discrete Mathematics and its Applications (Boca Raton). Chapman & Hall/CRC, Boca Raton, FL, 2005.

[17] A. Hulpke. Computing conjugacy classes of elements in matrix groups, *J. Algebra* **387** (2013), 268-286.

[18] Peter Kleidman and Martin Liebeck *The subgroup structure of the finite classical groups*, Cambridge University Press: Cambridge, 1990.

[19] M. W. Liebeck and J. Saxl. On the orders of maximal subgroups of the finite exceptional groups of Lie type, *Proc. London Math. Soc.* **55** (1987), 299–330.

[20] V. D. Mazurov. Minimal permutation representations of finite simple classical groups. Special linear, symplectic, and unitary groups, *Algebra Logika* **32** No. 3 (1993), 267–287.

[21] V. D. Mazurov and V. A. Vasilyev. Minimal permutation representations of finite simple orthogonal groups, *Algebra Logika* **33** No. 6 (1994), 603–627.

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

[23] M. Neunhöffer and Á. Seress. A data structure for a uniform approach to computations with finite groups. In *ISSAC '06: Proceedings of the 2006 international symposium on Symbolic and algebraic computation*, ACM, New York, 2006, 254–261.

[24] Max Neunhöffer and Ákos Seress *et al.* recog - methods for constructive recognition, `www-groups.mcs.st-and.ac.uk/~neunhoef/Computer/Software/Gap/recog.html`.

[25] W. H. Patton, *The minimum index for subgroups in some classical groups: a generalization of a theorem of Galois*, PhD Thesis, U. of Illinois at Chicago Circle, 1972.

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

[27] W. R. Unger. Computing the character table of a finite group, *J. Symbolic Comput.* **41** No. 8 (2006), 847-862.

[28] A. V. Vasilyev. Minimal permutation representations of finite simple exceptional groups of types G_2 and F_4, *Algebra Logika* **35** No. 6 (1996), 663–684.

[29] A. V. Vasilyev. Minimal permutation representations of finite simple exceptional groups of types E_6, E_7, and E_8, *Algebra Logika* **36** No. 5 (1997), 518–530.

[30] A. V. Vasilyev. Minimal permutation representations of finite simple exceptional twisted groups, *Algebra Logika* **37** No. 1 (1998), 9–20

[31] R. A. Wilson *et al.* Atlas of Finite Group Representations, `brauer.maths.qmul.ac.uk/Atlas`.