PRACTICAL COMPUTATION WITH LINEAR GROUPS
OVER INFINITE DOMAINS

A. S. DETINKO AND D. L. FLANNERY

Abstract. We survey recent progress in computing with finitely generated linear groups over infinite fields, describing the mathematical background of a methodology applied to design practical algorithms for these groups. Implementations of the algorithms have been used to perform extensive computer experiments.

1. Introduction

1.1. Motivation. Linear groups (synonymously, matrix groups) have been studied from the beginning of group theory. Matrices afford a convenient representation of groups that frequently arise in algebra, geometry, number theory, topology, and theoretical physics. Enhancements of technology and computer algebra systems have initiated a new phase in this classical subject, concerned with the design and implementation of algorithms for practical computation.

Computing with matrix groups over finite fields is now well-established [29]. The situation for linear groups over infinite domains is less advanced. Consequently we are motivated to obtain efficient methods, algorithms, and software for computing in this class of groups.

1.2. Representing linear groups in a computer. Input to any algorithm should be a finite set. Thus, in the first instance, we consider finitely generated linear groups. Certain linear groups that are not finitely generated can still be designated by a finite set—say, of polynomials, in the case of linear algebraic groups. Whereas an arbitrary linear group need not be finitely generated or algebraic, these are two major classes covering many applications.

Finitely generated linear groups are amenable to symbolic computation. Let $\mathbb{F}$ be a field of characteristic $p \geq 0$, and suppose that $G = \langle S \rangle$ where $S = \{g_1, \ldots, g_r\} \subseteq \text{GL}(n, \mathbb{F})$. Then $G$ is defined over a finitely generated extension of the prime subfield of $\mathbb{F}$. The classification of such field extensions implies that $G$ is a subgroup of $\text{GL}(n, \mathbb{L})$, where $\mathbb{L}$ is a finite degree extension of $\mathbb{F}(x_1, \ldots, x_m)$, $\mathbb{P}$ is a number field or finite field $\mathbb{F}_q$ of size $q$ for some $p$-power $q$, and the $x_i$ are algebraically independent indeterminates.

2010 Mathematics Subject Classification. Primary 20-02; secondary 20G15, 20H20, 68W30.

Key words and phrases. Matrix group, algorithm, computation, decidable problem.
This means that essentially we only have to deal with the aforementioned categories of fields. All of these are supported by the computer algebra system Magma [5].

We could restrict the ground domain to the subring \( R \subseteq \mathbb{F} \) generated by all entries of the \( g_i \) and \( g_i^{-1} \). After replacing the original field by such a ring, we apply congruence homomorphism techniques to transfer computing over \( R \) to computing over a quotient ring \( R/\rho \). If \( \rho \) is a maximal ideal then \( R/\rho \) is a finite field, and in that event the computational complexity is ameliorated by avoiding work over an infinite ring. We also gain access to the machinery for matrix groups over finite fields. See [20, Section 2] for details.

1.3. Properties of linear groups. We rely on classical theory of linear groups [21, 34, 36]. Two basic properties are crucial in our endeavours. One of these provides background for the computational methods; the other steers our overall strategy.

First, we recall that each finitely generated linear group \( G \) is residually finite. Moreover, \( G \) is ‘approximated’ by matrix groups of the same degree over finite fields. This approximation is effected by congruence homomorphisms \( \varphi_\rho : \text{GL}(n, R) \to \text{GL}(n, R/\rho) \). Since each non-zero element of \( R \) is missing from at least one ideal, and \( R/\rho \) is a finite field if we choose \( \rho \) to be a maximal ideal, the congruence images \( \varphi_\rho(G) \) realize the finite approximation.

A famous result of J. Tits [35] asserts that each finitely generated linear group over a field either is solvable-by-finite (virtually solvable), or contains a free non-abelian subgroup. The Tits alternative thereby divides finitely generated linear groups into two very different classes which require separate treatment.

2. Computing with virtually solvable groups

2.1. Method of finite approximation. Our techniques for computing with solvable-by-finite groups are broad-based and uniform, enabling us to solve a range of problems by similar algorithms. Underlying these features are deep results about the congruence subgroup \( G_\rho := G \cap \ker \varphi_\rho \).

**Theorem 2.1.** There exist maximal ideals \( \rho \) of \( R \) such that

(i) All torsion elements of \( G_\rho \) are unipotent. In particular, \( G_\rho \) is torsion-free if \( \text{char } R = 0 \).

(ii) If \( G \) is solvable-by-finite then \( G_\rho \) is unipotent-by-abelian as long as one of the following holds: \( \text{char } R > n ; \text{char } R = 0 \) and \( \text{char } (R/\rho) > n \); \( R \) is a Dedekind domain of characteristic zero and \( p \in \rho \setminus \rho^{p-1} \) for some odd prime \( p \).

See [36, Chapter 4] or [20, Section 2] for a proof of Theorem 2.1 (i). Proofs, and extra conditions on \( R \) and \( \rho \) guaranteeing the outcome in Theorem 2.1 (ii), are given in [37].
Our method begins by selecting $\rho$ according to the strictures of Theorem 2.1, and computing the congruence image $\varphi_\rho(G) \leq \text{GL}(n, R/\rho)$. Then we examine the structure of $G_\rho$. We call $\varphi_\rho$ for $\rho$ as in Theorem 2.1 a $W$-homomorphism. Algorithms to compute $W$-homomorphisms $\varphi_\rho$ and their corresponding congruence images $\varphi_\rho(G)$ were developed in [18, 20]. These compute normal generators of $G_\rho$, i.e., a finite set $N \subseteq G$ such that $G_\rho = \langle N \rangle^G$. The set $N$ is found by means of a presentation of $\varphi_\rho(G)$, computed using algorithms for matrix groups over finite fields [4, 29]. For our purposes, any relevant information about $G_\rho$ can be deduced from $N$; the full normal closure $\langle N \rangle^G$ is not needed.

2.2. Recognizing the type of a matrix group. Armed with practical methods, we proceed to the development of algorithms. Given $S \subseteq \text{GL}(n, \mathbb{F})$ we must first recognize the ‘type’ of $G = \langle S \rangle$. Once this is done, $G$ can be investigated using tools that are most appropriate for the group type. Below we note algorithms to recognize the type of $G$ (each of which requires selection of a single $W$-homomorphism $\varphi_\rho$). These algorithms additionally justify that the relevant problems are decidable for finitely generated linear groups over infinite fields.

2.2.1. Finiteness. In characteristic zero, $G$ is finite if and only if $G_\rho = \langle N \rangle^G = 1$. If $\text{char} \mathbb{F} = p > 0$ then finiteness testing turns on whether $G_\rho$ is a $p$-group, i.e., unipotent. See [20, Section 4].

2.2.2. Virtual solvability and other properties. We can recognize whether $G$ is solvable-by-finite: a computational realization of the Tits alternative. For this it is enough to test whether $G_\rho$ is unipotent-by-abelian, i.e., conjugate to a block-triangular group with all main diagonal blocks abelian. This test is carried out using manipulations with the enveloping algebra of $G_\rho$ over $\mathbb{F}$, as explained in [18, Section 3]. Although it decides whether a finitely generated linear group contains a free non-abelian subgroup, our algorithm does not construct one.

Algorithms to test whether $G$ is solvable, (virtually) nilpotent, abelian-by-finite, or central-by-finite. use a mix of ideas similar to the above [18, Section 5].

2.3. Investigating the structure of linear groups.

2.3.1. Finite groups. If $G$ is found to be finite then we can obtain an isomorphic copy over a finite field $\mathbb{F}_q$. In characteristic zero, $G \cong \varphi_\rho(G)$ for any $W$-homomorphism $\varphi_\rho$; in positive characteristic, repeated selection of $\rho$ may be needed to get an isomorphism $\varphi_\rho$ [20, Section 4.3]. Algorithms for matrix groups over finite fields may then be applied to $\varphi_\rho(G) \leq \text{GL}(n, q)$ to answer questions about the original group $G$. 

2.3.2. Solvable groups. Linear groups play a central role in the theory of infinite solvable groups. However, in designing algorithms for solvable linear groups we encounter serious obstacles, such as lack of decidability of various problems [25, Chapter 9]. To further illustrate this point, we make a comparison with polycyclic groups. Virtually polycyclic groups are finitely generated and \( \mathbb{Z} \)-linear. On the other hand, finitely generated (virtually) solvable linear groups need not be finitely presentable, they might have subgroups that are not finitely generated, and they do not satisfy the maximal condition on subgroups. Computing becomes viable with groups of finite Prüfer rank, which are solvable-by-finite and \( \mathbb{Q} \)-linear. Hence, we can test whether a finitely generated linear group \( G \) over a number field \( \mathbb{F} \) has finite rank. Furthermore, if \( G \) is (virtually) solvable then we can: compute the torsion-free rank (Hirsch number) of \( G \), and bounds on its Prüfer rank; test whether \( |G : H| \) is finite, for a finitely generated subgroup \( H \) of \( G \); construct a generating set of the completely reducible part of \( G \) (this includes testing whether \( G \) is completely reducible or unipotent). More generally, these algorithms work for solvable-by-finite groups \( G \) over any field, albeit with qualifications on \( G \) in positive characteristic. The papers [18, 19] contain lengthier discussion of the above.

Nilpotent-by-finite linear groups are more computationally tractable. Algorithms for these groups are given in [10] and [18, Section 5]. Computing with polycyclic linear groups is a separate topic (see, e.g., [2, 3]) beyond the remit of our survey.

2.3.3. Implementation. Many of our algorithms for virtually solvable groups were developed jointly with Eamonn O’Brien. Implementations are available in Magma; see [17]. Experimental results are reported in [18, Section 6], [19, Section 4.5], and [20, Section 5].

3. Dense and arithmetic groups

The methods of Section 2 could be developed further. However, to move beyond virtually solvable groups, new ideas are required.

Each linear group \( H \) is contained in an algebraic group, with the Zariski closure of \( H \) being the ‘smallest’ such overgroup. We will suppose that \( H \) is dense (in the Zariski topology) subgroup of an algebraic group. Note that an algorithm to compute the Zariski closure of a finitely generated linear group is given in [9].

The most interesting case is \( \mathbb{Q} \)-groups \( \mathcal{G} \leq \text{GL}(n, \mathbb{C}) \), i.e., \( \mathcal{G} \) is defined by a set of polynomials with coefficients in \( \mathbb{Q} \). For a subring \( R \subseteq \mathbb{C} \), denote \( \mathcal{G} \cap \text{GL}(n, R) \) by \( \mathcal{G}(R) \). Recall that \( H \leq \mathcal{G}(\mathbb{Q}) \) is arithmetic if \( H \cap \mathcal{G}(\mathbb{Z}) \) has finite index in \( H \) and in \( \mathcal{G}(\mathbb{Z}) \). In particular, finite index subgroups of \( \mathcal{G}(\mathbb{Z}) \) are arithmetic. Arithmetic groups are finitely generated and dense. If \( H \leq \mathcal{G}(\mathbb{Z}) \) is dense but not arithmetic, then we call \( H \) a thin matrix group (after [31]). A major open problem is testing whether finitely generated subgroups of \( \mathcal{G}(\mathbb{Z}) \) are arithmetic. In [11] we provide an algorithm (implemented in
Magma) to test arithmeticity when $G$ is solvable; showing that the problem is decidable with this proviso. The algorithm computes a generating set of an arithmetic subgroup in $G(\mathbb{Z})$, compares its Hirsch number with that of the input $H \leq G(\mathbb{Q})$, and tests integrality of $H$.

3.1. Density and computing with linear groups. Most linear groups are not virtually solvable [1, 22]. So we cannot expect to handle every finitely generated linear group $H$ that is not virtually solvable by a single uniform method. Selecting one ideal at a time might not suffice for all problems.

We are viewing $H$ as a subgroup of some algebraic Q-group $G$, which may be assumed semisimple by a standard reduction [8, Chapters 3 and 4]. Since $H$ should be dense in $G$, density testing is a preliminary task. A deterministic algorithm to test density of $H$ is given in [30], together with a Monte-Carlo algorithm that tests density of $H \leq G(\mathbb{Z})$ for $G = \text{SL}(n, \mathbb{C})$ or $\text{Sp}(n, \mathbb{C})$; see also [13, Section 3.2]. These algorithms have been implemented in GAP [23] (see [14]).

3.2. From finite to strong approximation. We have expanded the congruence homomorphism methodology to cover dense subgroups. For certain $G$, and $H \leq G(\mathbb{Z})$ dense in $G$, a celebrated result known as the strong approximation theorem [27, Window 9] enables us to compute all congruence quotients of $H$ modulo primes. Both $\text{SL}(n, \mathbb{C})$ and $\text{Sp}(n, \mathbb{C})$ are suitable examples of such $G$; from now on $G$ stands for either of these two groups. Strong approximation implies that if $H \leq G(\mathbb{Z})$ is dense then $\varphi_p(H) = \varphi_p(G(\mathbb{Z}))$ for all but finitely many primes $p$. Denote the set of these exceptional primes by $\Pi(H)$. We have developed practical algorithms to compute $\Pi(H)$, thus realizing strong approximation computationally; see [13, Section 3.2], [15, 16]. Our methods for computing $\Pi(H)$ draw on classifications of maximal subgroups in $\text{SL}(n, p)$ and $\text{Sp}(n, p)$, and subgroups of $\text{GL}(n, p)$ with a known transvection. Actually, once we have $\Pi(H)$ we can find all congruence quotients of $H$ [15, 16].

3.3. From density to arithmeticity. Let $n > 2$ and $H \leq G(\mathbb{Z})$ be dense. Then $H$ lies in a unique ‘minimal’ arithmetic group $\text{cl}(H)$, namely the intersection of all arithmetic groups in $G(\mathbb{Z})$ containing $H$. Algorithms for arithmetic subgroups of $G(\mathbb{Z})$ can therefore be used to study dense subgroups as well.

We gain much mileage from the fact that $\Gamma_n := G(\mathbb{Z})$ has the congruence subgroup property: each arithmetic group $H$ in $\Gamma_n$ contains a principal congruence subgroup (PCS), which is the kernel of a congruence homomorphism $\varphi_m : \Gamma_n \rightarrow \text{GL}(n, \mathbb{Z}_m)$ for some $m$. Here $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$, and $m$ is called the level of the PCS. The maximal PCS of $H$ is unique, and its level $M = M(H)$ is defined to be the level of $H$. Similarly, for dense $H \leq \Gamma_n$, we assign $M(H)$ as the level of $\text{cl}(H)$. 
3.4. **Computing via the congruence subgroup property.** The bedrock of our method for computing with dense groups is the congruence subgroup property. It splits our method into two overlapping parts: finding $M(H)$, and computing in $\text{GL}(n, \mathbb{Z}_m)$.

3.4.1. **Computing the level.** The set $\pi(M)$ of prime divisors of $M(H)$ coincides with $\Pi(H)$, besides minor exceptions for $n = 3, 4$ and $p = 2$ (which are dealt with separately); see [13, Section 2.4]. Thus, the strong approximation algorithms cited in Section 3.2 yield $\pi(M)$. We can also compute the largest power of $p$ dividing $M(H)$ for each $p \in \pi(M)$. These two steps constitute the procedure LevelMaxPCS, which accepts $\Pi(H)$ and a generating set $S$ of a dense group $H \leq \Gamma_n$, and returns its level.

3.4.2. **Computing with groups over $\mathbb{Z}_m$.** Algorithms for subgroups of $\text{GL}(n, \mathbb{Z}_m)$ have intrinsic value. We reduce computing to the situations of matrix groups over finite fields, and groups of prime-power order. Two major steps in the reduction are as follows. Say $m = p_1^{k_1} \ldots p_t^{k_t}$ where the $p_i$ are distinct primes and all $k_i$ are non-zero. Then (essentially by the Chinese Remainder Theorem)

\[
\begin{align*}
\text{(i)} & \quad \text{GL}(n, \mathbb{Z}_m) \cong \text{GL}(n, \mathbb{Z}_{p_1^{k_1}}) \times \cdots \times \text{GL}(n, \mathbb{Z}_{p_t^{k_t}}) \\
\text{(ii)} & \quad \text{GL}(n, \mathbb{Z}_{p^k})/K \cong \text{GL}(n, p), \quad \text{where} \ K = \{h \in \text{GL}(n, \mathbb{Z}_{p^k}) \mid h \equiv 1_n \mod p^{k-1}\} \text{ is a } p\text{-group.}
\end{align*}
\]

We also use the fact $K \cap G$ almost always does not have a proper supplement in $G$, for $G = \text{SL}(n, \mathbb{Z}_{p^k})$ or $\text{Sp}(n, \mathbb{Z}_{p^k})$ [13, Theorem 2.5].

3.5. **Algorithms for arithmetic subgroups.** Let $H \leq \Gamma_n$ be arithmetic. In the application of Section 3.4 to designing algorithms for $H$, the main steps are LevelMaxPCS, and computing with matrix groups over finite rings $\mathbb{Z}_m$. One example is the membership test IsIn$(g, H)$ which determines whether $g \in \Gamma_n$ is in $H$; it merely checks whether $\varphi_M(g) \in \varphi_M(H)$. We emphasize that our results imply decidability of membership testing in arithmetic groups in $\Gamma_n$. An associated algorithm computes $|\Gamma_n : H|$. Although the index could be calculated in the congruence image, i.e., as $|\varphi_M(\Gamma_n) : \varphi_M(H)|$, in practice $|\Gamma_n : H|$ is found as a byproduct of computing $M$ [13, Section 2.4.2] (see [12, Section 6] and [13, Section 4]). Since membership testing and computing the index are both decidable, an arithmetic group $H \leq \Gamma_n$ is ‘explicitly given’ as per [24]. Other notable algorithmic problems for arithmetic subgroups are therefore decidable too.

3.6. **Further computation with arithmetic subgroups.**

3.6.1. **Structural analysis.** Arithmetic groups are matrix groups over rings, so their (sub)normal subgroups are of interest. The procedure IsSubnormal$(H)$ tests whether $H$ is subnormal in $\Gamma_n$; Normalizer$(H)$ computes a generating set of the normalizer of $H$ in $\Gamma_n$; NormalClosure$(B)$ computes a generating set of the normal closure in $\Gamma_n$ of the group generated by $B \subset \Gamma_n$. Other
procedures are given in [12, Section 3.2]. Many more algorithms could be developed along these lines.

3.6.2. The orbit-stabilizer problem. Let \( n > 2 \) and \( H \leq \text{SL}(n, \mathbb{Z}) \) be arithmetic. Given \( u, v \in \mathbb{Q}^n \), the procedure \texttt{Orbit}(\( u, v \)) tests whether there is \( g \in \text{SL}(n, \mathbb{Z}) \) such that \( gu = v \), and computes \( g \) if it exists. \texttt{Stabilizer}(\( H, u \)) returns the (finitely generated) stabilizer of \( u \) in \( H \). Both procedures solve the related orbit and stabilizer problems for the congruence image over \( \mathbb{Z}_M \) and for the maximal PCS in \( H \) acting on \( \mathbb{Q}^n \). The outputs are then combined. See [12, Section 4].

3.7. Experiments. The algorithms of this section are joint work with Alexander Hulpke. Below we review some experiments illustrating our GAP implementation of the algorithms and their practicality; see [13, 15, 16] for more.

3.7.1. Integral representations of the fundamental group \( \langle x, y, z | xzx^{-1} = xy, zyz^{-1} = yxy \rangle \) of the figure-eight knot complement are constructed in [26]. For non-zero \( T \in \mathbb{Z} \), let \( \beta_T(x) = X_T \) and \( \beta_T(y) = Y_T \) where

\[
X_T = \begin{bmatrix} -1 + T^3 & -T & T^2 \\ 0 & -1 & 2T \\ -T & 0 & 1 \end{bmatrix}, \quad Y_T = \begin{bmatrix} -1 & 0 & 0 \\ -T^2 & 1 & -T \\ T & 0 & -1 \end{bmatrix}.
\]

Then \( \beta_T \) is a homomorphism and \( \beta_T(\langle x, y \rangle) \leq \text{SL}(3, \mathbb{Z}) \) is arithmetic. Construction of these representations was motivated by long-standing problems; such as the conjecture that each arithmetic group in \( \text{SL}(n, \mathbb{Z}) \) has a 2-generator finite index subgroup. The conjecture has been settled affirmatively [28]. Still, the subgroups \( \langle X_T, Y_T \rangle \) merit closer scrutiny. Earlier attempts to compute \( |\text{SL}(3, \mathbb{Z}) : \langle X_T, Y_T \rangle| \) were stymied by the fact that this index may be arbitrarily large. We were able to compute indices using our algorithms (see [13, Section 4.1]). For example, let \( T = 100 \); the index \( 2^{12}3^55^{25}7^{143}31^{2}67^{1783} \) and level \( 2^{17}3^{6}5^{2} \) were found in 892.6 seconds.

3.7.2. A second family of test groups comes from applications in theoretical physics. Let \( G(d, k) = \langle U, T \rangle \) where

\[
U = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ d & d & 1 & 0 \\ 0 & -k & -1 & 1 \end{bmatrix}, \quad T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\]

For fourteen pairs \( d, k \) of integers, \( G(d, k) \leq \text{Sp}(4, \mathbb{Z}) \) is the monodromy group of a generalized hypergeometric ordinary differential equation associated to Calabi-Yau threefolds. Seven of these groups are arithmetic, while the rest are thin [32, 33]. To investigate the latter, one could attempt to construct arithmetic groups in \( \text{Sp}(4, \mathbb{Z}) \) containing them [6]. We successfully computed \( \text{cl}(G(d, k)) \) for the seven thin groups [13, Table 3]; e.g., it took 25 seconds to find the level \( 2^{5}3^{2} \) and the index \( 2^{17}3^{6}5^{2} \) of \( G(12, 7) \).
4. Where to next?

We outline avenues for future research.

New methods and algorithms for algebraic groups and Lie algebras would have an impact on computing with virtually solvable groups. Despite significant progress (cf. Section 2), key algorithmic questions are still unresolved. One of these is membership testing. This problem is known to be decidable for groups of finite rank. The main challenge is handling the unipotent radical, which is a torsion-free nilpotent group that may not be finitely generated. Lie algebra methods due to P. Hall, and computing in ambient solvable algebraic groups, are possible approaches. These are similarly promising in the design of algorithms for structural analysis of virtually solvable linear groups. We also expect a number of new algorithms for computing with (virtually) nilpotent and (virtually) polycyclic linear groups.

Methods based on algebraic group techniques will be productive in applications to non-virtually solvable groups (cf. Section 3). Arithmeticity testing is open in general, even for subgroups of $\text{SL}(n, \mathbb{Z})$. Indeed, it is not known whether the problem is decidable. Computing generating sets and presentations of arithmetic subgroups are supplementary problems (cf. [8, Chapter 6], [7]). Construction of free subgroups would aid in the study of matrix groups that are not virtually solvable; large free subgroups, i.e., those that are dense in the Zariski closure, are especially useful. Testing freeness of finitely generated linear groups is yet another priority.

We await breakthroughs that apply computational methods to the solution of hard problems in group theory, other areas of mathematics, and farther afield (cf. Section 3.7). Here we point to computing linear representations of finitely presented groups: in contrast to the same problem for finite groups, much remains to be done.

Acknowledgments. We are indebted to our collaborators Willem de Graaf, Alexander Hulpke, and Eamonn O’Brien. We also thank Mathematisches Forschungsinstitut Oberwolfach, and the International Centre for Mathematical Sciences, UK, for hosting our visits under their ‘Research in Pairs’ and ‘Research in Groups’ programmes. A. S. Detinko is supported by a Marie Skłodowska-Curie Individual Fellowship grant (Horizon 2020, EU Framework Programme for Research and Innovation).

References

1. R. Aoun, Random subgroups of linear groups are free, Duke Math. J. (1) 160 (2011), 117–173.
2. B. Assmann and B. Eick, Computing polycyclic presentations for polycyclic rational matrix groups, J. Symbolic Comput. (6) 40 (2005), 1269–1284.
3. B. Assmann and B. Eick, Testing polycyclicity of finitely generated rational matrix groups, Math. Comp. 76 (2007), 1669–1682.
4. H. Bäärnhielm, D. Holt, C. R. Leedham-Green, and E. A. O’Brien, A practical model for computation with matrix groups, J. Symbolic Comput. 68 (2015), 27–60.
5. W. Bosma, J. Cannon, and C. Playoust, The Magma algebra system. I. The user language, J. Symbolic Comput. (3-4) 24 (1997), 235–265.
6. Y. Chen, Y. Yang, and N. Yui, Monodromy of Calabi-Yau differential equations (with an appendix by Cord Erdenberger), J. Reine Angew. Math. 616 (2008), 167–203.
7. R. Coulangeon, G. Nebe, O. Braun, and S. Schönnenbeck, Computing in arithmetic groups with Voronoi’s algorithm, J. Algebra (1) 435 (2015), 263–285.
8. W. de Graaf, Computation with linear algebraic groups (Chapman & Hall/CRC, 2017).
9. H. Derksen, E. Jeandel, and P. Koiran, Quantum automata and algebraic groups, J. Symbolic Comput. (3-4) 39 (2005), 357–371.
10. A. S. Detinko and D. L. Flannery, Algorithms for computing with nilpotent matrix groups over infinite domains, J. Symbolic Comput. (1) 43 (2008), 8–26.
11. A. S. Detinko, D. L. Flannery, and W. A. de Graaf, Integrality and arithmeticity of solvable linear groups, J. Symbolic Comput. 68 (2015), 138–145.
12. A. S. Detinko, D. L. Flannery, and A. Hulpke, Algorithms for arithmetic groups with the congruence subgroup property, J. Algebra 421 (2015), 234–259.
13. A. S. Detinko, D. L. Flannery, and A. Hulpke, Zariski density and computing in arithmetic groups, Math. Comp. 87 (2018), 967–986.
14. A. S. Detinko, D. L. Flannery, and A. Hulpke, GAP functionality for Zariski dense groups, Oberwolfach Preprints, OWP 2017-22.
15. A. S. Detinko, D. L. Flannery, and A. Hulpke, Algorithms for experimenting with Zariski dense subgroups, Exp. Math. to appear.
16. A. S. Detinko, D. L. Flannery, and A. Hulpke, The strong approximation theorem and computing with linear groups, preprint (2018).
17. A. S. Detinko, D. L. Flannery, and E. A. O’Brien, http://magma.maths.usyd.edu.au/magma/handbook/matrix_groups_over_infinite_fields
18. A. S. Detinko, D. L. Flannery, and E. A. O’Brien, Algorithms for the Tits alternative and related problems, J. Algebra 344 (2011), 397–406.
19. A. S. Detinko, D. L. Flannery, and E. A. O’Brien, Algorithms for linear groups of finite rank, J. Algebra 393 (2013), 187–196.
20. 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.
21. J. Dixon, The structure of linear groups (Van Nostrand Reinhold, London 1971).
22. D. B. A. Epstein, Almost all subgroups of a Lie group are free, J. Algebra 19 (1971), 261–262.
23. The GAP Group, GAP – Groups, Algorithms, and Programming, Version 4.8.7; 2017, http://www.gap-system.org
24. F. Grunewald and D. Segal, Some general algorithms. I. Arithmetic groups, Ann. of Math. (3) 112 (1980), 531–583.
25. J. C. Lennox and D. J. S. Robinson, The theory of infinite soluble groups (OUP, Oxford 2004).
26. D. D. Long and A. W. Reid, Small subgroups of SL(3, Z), Exp. Math. (4) 20 (2011), 412–425.
27. A. Lubotzky and D. Segal, Subgroup growth, Progress in Mathematics, Vol. 212, (Birkhäuser, Basel 2003).
28. C. Meiri, Generating pairs for finite index subgroups of SL(n, Z), J. Algebra 470 (2017), 420–424.
29. E. A. O’Brien, Algorithms for matrix groups, in Groups St Andrews 2009 in Bath. Vol. 2, (C. M. Campbell et al., eds.), London Math. Soc. Lecture Note Ser. 388 (CUP, Cambridge 2011), 297–323.
30. I. Rivin, Large Galois groups with applications to Zariski density, http://arxiv.org/abs/1312.3009v4
31. P. Sarnak, Notes on thin matrix groups, in Thin groups and superstrong approximation, Math. Sci. Res. Inst. Publ. 61, (CUP, Cambridge 2014), 343–362.
32. S. Singh, Arithmeticty of four hypergeometric monodromy groups associated to Calabi-Yau threefolds, Int. Math. Res. Notices (18), 2015 (2015), 8874–8889.
33. S. Singh and T. Venkataramana, Arithmeticty of certain symplectic hypergeometric groups, Duke Math. J. (3) 163 (2014), 591–617.
34. D. A. Suprunenko, Matrix groups, Translations of Mathematical Monographs, Vol. 45 (AMS, Providence 1976).
35. J. Tits, Free subgroups in linear groups, J. Algebra 20 (1972), 250–270.
36. B. A. F. Wehrfritz, Infinite linear groups (Springer-Verlag, New York 1973).
37. B. A. F. Wehrfritz, Conditions for linear groups to have unipotent derived subgroups, J. Algebra 323 (2010), 3147–3154.

School of Computer Science, University of St Andrews, North Haugh, St Andrews KY16 9SX, UK
E-mail address: ad271@st-andrews.ac.uk

School of Mathematics, Statistics and Applied Mathematics, National University of Ireland Galway, Galway H91TK33, Ireland
E-mail address: dane.flannery@nuigalway.ie