Ore's Conjecture and Computational Group Theory

Final Degree Dissertation
Degree in Mathematics

Xabier de Juan Soriano

Supervisor:
Matteo Vannacci

Leioa, June 22, 2023

Contents

Symbols

S_n	Symmetric group on n elements.
A_n	Alternating group on n elements.
$\lvert g\rvert$	Order of the element g.
g^h	Conjugate $h^{-1}gh$.
$\langle S\rangle^G$	Normal closure of a subset S of a group G.
$\mathrm{Cl}_G(g)$	Conjugacy class of g on a group G.
$C_G(S)$	Centralizer of a subset S of a group G.
$K(G)$	Set of all commutators of a group G.
G'	Derived subrgoup of G.
$x\varphi$ or $(x)\varphi$	Image of x by the homomorphism φ.
fg	The composition of maps where we apply first f and then g.
$\mathbb{C}G$	$\mathbb{C}$-group algebra of the group G.
$\operatorname{Irr} G$	Set of complex irreducible characters of G.
$\mathbb{F}_q$	Finite field of prime-power order q.
$k^\times$	Group of units of the field k.
A^T	Transpose of the matrix A.
$\operatorname{tr} A$	Trace of the matrix/linear map A.
$A_{i,:}$	i-th row of the matrix A.
$v[i]$	i-th component of the vector v.
$M_n(k)$	Algebra of $n \times n$ matrices over a field k.
I_n	$n \times n$ identity matrix.
$\operatorname{diag}(\lambda_1, \dots, \lambda_n)$	Square diagonal matrix with the elements $\lambda_1, \dots, \lambda_n$ on the main diagonal.
$M_{\mathcal{B}}(f)$	Matrix representation by rows of the linear map f w.r.t. the basis $\mathcal{B}$.
V^*	Dual space of V.
(n, m)	Greatest common divisor of n and m.
$\equiv_n$	Congruence mod n.
1_A	Identity map on the set A.

Introduction

A group is said to be *simple* if its only normal subgroups are the trivial one and the group itself. The finite simple groups play a crucial role in understanding finite groups as they can be thought of as their "basic building blocks".

The study of simple (non-abelian) groups can be dated back to the early 19th century, when Évariste Galois studied the simple group A_5. He also constructed the simple groups $\mathrm{PSL}_2(p)$ for primes $p \geqslant 5$. This construction can be found in his last letter, addressed to his friend Auguste Chevalier[1].

There was not much progress in the theory of finite simple groups until the end of the 19th century. In 1870, Camille Jordan constructed in [Jor70] four families of simple matrix groups over the fields $\mathbb{F}_p$ of prime order. The beginning of the 20th century saw the creation of a well-developed theory of the finite groups. During the early 1960s, serious efforts to classify finite simple groups began. However, this task turned out to be much more challenging than what some people initially anticipated. Finally, in 2004 the classification of all finite simple groups was completed [Wil09]. The proof, which involved the collaboration of about 100 authors, consists of tens of thousands of pages spread over a hundred journal articles. This result is considered a milestone of twentieth-century Mathematics.

Classification Theorem for Finite Simple Groups. *Every finite simple group is isomorphic to one of the following:*

- *a member of one of three infinite classes:*

  (i) *a cyclic group C_p of prime order p;*

  (ii) *an alternating group A_n, when $n \geqslant 5$;*

  (iii) *a group of Lie type:*

[1]It was written on May 29, 1832, one day before his fatal duel, at the age of 20. It is considered his testament as a mathematician. Galois asks Chevalier to publicly ask Jacobi or Gauss to give their opinion, not on the truth, but on the importance of the theorems that he has found in the letter, and to have the letter printed in the *Revue encyclopédique*. The letter was published in September 1832.

- *a classical group:*

linear:	$\mathrm{PSL}_n(q)$, $n \geqslant 2$, *except* $\mathrm{PSL}_2(2)$ *and* $\mathrm{PSL}_2(3)$;
unitary:	$\mathrm{PSU}_n(q)$, $n \geqslant 3$, *except* $\mathrm{PSU}_3(2)$;
symplectic:	$\mathrm{PSp}_{2n}(q)$, $n \geqslant 2$, *except* $\mathrm{PSp}_4(2)$;
orthogonal:	$\mathrm{P\Omega}_{2n+1}(q)$, $n \geqslant 3$, q *odd*;
	$\mathrm{P\Omega}^{\varepsilon}_{2n}(q)$, $n \geqslant 4$, $\varepsilon \in \{+,-\}$

  where q is a prime power;

- *an exceptional group of Lie type,*

$$G_2(q), q \geqslant 3; F_4(q); E_6(q); {}^2E_6(q); {}^3D_4(q); E_7(q); E_8(q)$$

  where q is a prime power, or

$$^2B_2(2^{2n+1}); {}^2G_2(3^{2n+1}); {}^2F_4(2^{2n+1})$$

  where $n \geqslant 1$, or the Tits group ${}^2F_4(2)'$.

- *one of the 26 sporadic simple groups:*

  (i) *a Mathieu group* M_{11}*,* M_{12}*,* M_{22}*,* M_{23}*,* M_{24}*;*

  (ii) *a Leech lattice group* Co_1*,* Co_2*,* Co_3*,* McL*,* HS*,* Suz*,* J_2*;*

  (iii) *a Fischer group* Fi_{22}*,* Fi_{23}*,* Fi'_{24}*;*

  (iv) *a Monstrous group* $\mathbb{M}$*,* $\mathbb{B}$*,* Th*,* HN*,* He*;*

  (v) *a pariah* J_1*,* J_3*,* J_4*,* O'N*,* Ly*,* Ru.

In a group, the *commutator* of two elements x and y is defined as

$$[x,y] = x^{-1}y^{-1}xy.$$

In 1951, Øystein Ore dealt with commutators in [Ore51], where he proved that, if $n \geqslant 5$, every permutation of A_n is a commutator of two permutations of S_n. He also affirmed that the proof could be extended to show that in A_n every element is a commutator. In the final part of the same article he stated the following: *"It is possible that a similar theorem holds for any simple group of finite order, but it seems that at present we do not have the necessary methods to investigate the question."* This has become known as *Ore's conjecture.*

Through different articles published throughout the 20th century, Ore's conjecture was established by many authors for different families of finite simple groups. For instance, Noboru Ito proved the conjecture for the simple alternating groups in [Ito51]. In [NPC84], Ore's conjecture was checked directly using computational methods for the sporadic groups.

Several authors obtained partial results for finite simple groups of Lie type [Mal14]. In 1998, an important breakthrough was made by Erich Ellers and Nikolai Gordeev in [EG98], where they established Ore's conjecture for groups of Lie type over a finite field $\mathbb{F}_q$, if $q \geqslant 8$. Finally, in 2009 the remaining cases (groups of Lie type over $\mathbb{F}_q$ with $q < 8$) were proved in [LOST]. Combining this and the classification of the finite simple groups, Ore's conjecture was finally proven.

Main Theorem. *If G is a finite non-abelian simple group, then every element of G is a commutator.*

In fact, in [LOST] a more general result was proved for the classical groups than Main Theorem. It was shown that in every quasisimple (perfect group G such that $G/Z(G)$ is simple) classical group every element is a commutator.

Now, we give a rough idea of the strategy used in [LOST] to prove the conjecture. A dichotomy is established between elements with "small centralizer" and the rest. For an element with small centralizer, Deligne-Lusztig theory and the theory of dual pairs and Weil characters of classical groups are used to show that such an element is a commutator.

On the other hand, for a fixed element whose centralizer is not small, the strategy consists of reducing to groups of Lie type of lower dimension and apply induction. For instance, for symplectic or orthogonal groups, it is possible to write such an element as a Jordan decomposition into several Jordan blocks. Therefore, this element lies in direct product of smaller symplectic or orthogonal groups. Hence, if one can inductively express each block as a commutator in a lower rank classical group, then the fixed element is a commutator. However, for the unitary groups, the inductive strategy using Jordan blocks does not work well, and therefore M. Liebeck et al. adopt a different approach.

This dissertation arises from [LOST, Lemma 3.1], which we state and name as Main Lemma. This lemma serves as the base of the induction of the proof of the Main Theorem.

Main Lemma. *Every element of each of the following groups is a commutator:*

(i) $\mathrm{Sp}_{2m}(2)$, $3 \leqslant m \leqslant 6$;

(ii) $\mathrm{Sp}_{2m}(3)$, $2 \leqslant m \leqslant 5$;

(iii) $\mathrm{SU}_3(q)$, $3 \leqslant q \leqslant 17$; $\mathrm{SU}_4(q)$, $q \leqslant 7$; $\mathrm{SU}_5(q)$, $q \leqslant 4$; $\mathrm{SU}_6(q)$, $q \leqslant 4$; $\mathrm{SU}_7(2)$;

(iv) $\Omega_n^{\pm}(2)$, $8 \leqslant n \leqslant 12$; $\Omega_n^{\pm}(3)$, $7 \leqslant n \leqslant 11$; $\Omega_7(5)$;

(v) *simply connected* $D_4(q)$, $q \leqslant 4$; ${}^2D_4(q)$, $q \leqslant 5$;

(vi) $E_6(2)$ *or simply connected* ${}^2E_6(2)$.

The proof of the Main Lemma is by direct computation in a computer, and most of the details are omitted in the article. Therefore, the goal of this dissertation is to reproduce such computations and understand some of the algorithms behind them.

It is assumed that the reader does not have any prior knowledge apart from what is taught in the Mathematics degree. Therefore, the first chapter is dedicated to the definitions of the classical groups, which in most cases arise as the quotients of some isometry groups of bilinear/sesquilinear forms on a vector space. Furthermore, we will provide the full proofs of the simplicity of two families of classical groups.

We start Chapter 2 proving Ore's criterion, which relates Ore's conjecture for a concrete group with the values of its ordinary character table. In order to apply this criterion to a particular group, one should know the complete character table of such a group. Hence, we will also present an algorithm for computing the character table of an arbitrary group. We will implement this algorithm in the GAP system [GAP22], which is a computational discrete algebra system, with particular emphasis on Computational Group Theory.

At last, we begin the final chapter by noting some generalizations of Ore's conjecture. Later, we establish Ore's conjecture for the alternating groups A_n when $n \geqslant 5$. Throughout the literature, in [KM05] among others, it is said that Ore's conjecture is established for the alternating groups in [Mil99], [Ito51] and [TL65]. The proof presented in [Mil99] is far from being a rigorous and complete, nonetheless we have been inspired by it in the proof we present in this dissertation. At the time of writing this dissertation we have not been able to find both articles [Ito51] and [TL65]. Also, we will present two tests that check if a group fulfills Ore's conjecture or not. One consists in applying Ore's criterion, whereas the other is a Las Vegas algorithm. We will also implement these tests in the GAP system [GAP22]. Finally, we will compare the performance of the different tests.

All the code written for this dissertation can be found in Appendix B or in the GitHub repository [Jua23].

Chapter 1

The classical groups

In this chapter we define and prove the simplicity of some of the *classical groups*, as they were named by Hermann Weyl in [Wey39]. These groups, in essence, are quotients of matrix groups by their centres.

The tale begins with *Her All-embracing Majesty* (quoting H. Weyl [Wey39, p. 136]), the *general linear group*, which consists of all invertible linear maps of a vector space on itself. All other classical groups arise as subgroups of the general linear group or as closely related quotient groups. These are: the symplectic and unitary groups, and the three families of orthogonal groups.

We follow mostly [Wil09, Ch. 3] and [Gro01]. Particularly, in Section 1.2 also [Lan02, Ch. XIII, §9] and [Con20].

1.1 Iwasawa's lemma

Before anything else, we recall some concepts of group actions. In this dissertation all actions will be *right actions*. Let G be a group acting on a set Ω. A *block* is a subset B of Ω such that for all $g \in G$, either $Bg = B$ or $Bg \cap B = \emptyset$. If G acts transitively on a set Ω that has more than one element, then we say that G acts *primitively* on Ω if the only blocks are the empty set, singletons and the whole Ω. These actions can be characterized in the following way: a group action is primitive if and only if for any $\omega \in \Omega$, the group $\mathrm{Stab}_G(\omega)$ is a maximal subgroup of G [Gro01, p. 3]. An action of G on Ω is said to be 2-*transitive* if for any pairs $(x, y), (x', y')$ of distinct elements of Ω, there exists $g \in G$ with $gx = x'$ and $gy = y'$. Finally, we remark that any 2-transitive action is primitive [Gro01, Proposition 0.2].

Our main tool for proving the simplicity of the classical groups will be Iwasawa's Lemma.

Lemma 1.1.1 (Iwasawa)**.** *Let G be a finite group acting faithfully and primitively on a set Ω. Assume the following:*

(i) *For some $\omega \in \Omega$, the group $\mathrm{Stab}_G(\omega)$ contains a normal abelian subgroup N whose conjugate subgroups generate G;*

(ii) *$G' = G$, i.e. G is perfect.*

Then G is simple.

Proof. Assume, for sake of contradiction, that there is a normal subgroup K such that $1 < K < G$. As G acts faithfully on Ω and $K \neq \{1\}$, K does not fix all the points of Ω. Therefore, there is $\omega_0 \in \Omega$ such that $K \nleqslant \mathrm{Stab}_G(\omega_0) =: H$. Since $K \lhd G$, HK is a subgroup of G. As G acts primitively on Ω, H is a maximal subgroup of G, and therefore, either $H = HK$ or $G = HK$. As $K \nleqslant H$, we conclude that $G = HK$.

Now, we claim that H contains a normal abelian subgroup, A, whose conjugate subgroups generate G. By hypothesis, we know that such property holds for some $\mathrm{Stab}_G(\omega)$, $\omega \in \Omega$. Recalling that the action is transitive, we have that for some $g \in G$, $\mathrm{Stab}_G(\omega)^g = H$. Then $A = N^g$ is a normal abelian subgroup of H. Also, clearly, the conjugates of A generate G.

Since $A \trianglelefteqslant H$, $AK \trianglelefteqslant HK = G$. Then for any $g \in G$, $A^g = g^{-1}Ag \subset g^{-1}(AK)g = AK$. Therefore, as $G = \langle A \rangle^G$, we conclude that $G = AK$. Finally, by the second isomorphism theorem, $G/K = AK/K \cong A/(A \cap K)$. So, G/K is abelian and consequently $G = G' \leqslant K$, which is a contradiction. □

When dealing with the classical groups, the subgroup N from Iwasawa's lemma contains a particular type of endomorphisms, the transvections. A *transvection* is an endomorphism f on a vector space V such that $\mathrm{rk}(f-1) = 1$ and $(f-1)^2 = 0$. One can show that the inverse of a transvection is again a transvection [Gro01, p. 7]. For any transvection f there exist $\varphi \in V^*$ and $v \in \ker\varphi$ such that $f\colon x \mapsto x + (x)\varphi v$ [CS99]. In this case, we use the notation $T_v(\varphi)$ to denote f.

1.2 Linear groups

Let V be an n-dimensional $\mathbb{F}_q$-vector space. Then $\mathrm{GL}(V)$ denotes the *general linear group* of V, the group of all invertible linear maps from V to itself. Fixing a basis of V yields an isomorphism between $\mathrm{GL}(V)$ and the group $\mathrm{GL}_n(q)$ of all invertible $n \times n$ matrices over the field $\mathbb{F}_q$. Since $\det(AB) = \det(A)\det(B)$, the map $\det\colon \mathrm{GL}_n(q) \to \mathbb{F}_q^\times$ is a group homomorphism and its kernel is called the *special linear group* $\mathrm{SL}_n(q)$, which consists of all the matrices of determinant 1. Now we define the *projective special linear group* $\mathrm{PSL}_n(q)$ as the quotient of $\mathrm{SL}_n(q)$ by its centre: $\mathrm{PSL}_n(q) = \mathrm{SL}_n(q)/Z(\mathrm{SL}_n(q))$. It is well known that $Z(\mathrm{SL}_n(q))$ is the set of scalar matrices with determinant 1. In Exercise A.1.1 we compute the orders of all of these groups.

After some preliminaries we will be able to apply Iwasawa's lemma to $\mathrm{PSL}_n(q)$. We let $\mathrm{SL}_n(q)$ act on the set Ω of 1-dimensional subspaces of

V in the natural way. By linear algebra, any basis can be transformed, up to scalar multiplication, to any other basis by a matrix of $\mathrm{SL}_n(q)$. In particular, the action is 2-transitive, and hence primitive. Moreover, it can be easily shown that the kernel of this action is $Z(\mathrm{SL}_n(q))$ [Gro01, p. 6]. Hence, $\mathrm{PSL}_n(q)$ acts[1] primitively and faithfully on Ω.

It is easy to check that the group $H = \mathrm{Stab}_{\mathrm{SL}_n(q)}(\langle(1, 0, \dots, 0)\rangle)$ consists of matrices whose first row is the vector $(\lambda, 0, \dots, 0)$ for any $\lambda \in \mathbb{F}_q^\times$. Now, we claim that the set N which consists of matrices of the form

$$\begin{pmatrix} 1 & 0_{n-1} \\ v_{n-1} & I_{n-1} \end{pmatrix},$$

where v_{n-1} denotes an arbitrary column vector with coefficients in $\mathbb{F}_q$, is a normal abelian subgroup of H. Let φ be the map from H to $\mathrm{GL}_{n-1}(q)$ which maps the element

$$\begin{pmatrix} \lambda & 0_{n-1} \\ v_{n-1} & B \end{pmatrix}$$

of H to B. It can be easily shown that in fact φ is a group epimorphism and that $\ker \varphi = N$, whence we get that $N \trianglelefteqslant H$. Furthermore, as $\ker \varphi \cong \mathbb{F}_q^{n-1}$, in particular N is abelian.

For $1 \leqslant i, j \leqslant n$ with $i \neq j$ and $\lambda \in \mathbb{F}_q^\times$ we denote by $E_{i,j}(\lambda)$ the matrix which differs from I_n by having λ in the (i, j)-th component instead of 0. We call such $E_{i,j}(\lambda)$ an *elementary matrix.* It is clear that $E_{i,j}(\lambda) \in \mathrm{SL}_n(q)$. If A is an $n \times n$ matrix, then the multiplication $E_{i,j}(\lambda)A$ corresponds to adding λ times the j-th row to the i-th row of A. In particular, the inverse of $E_{i,j}(\lambda)$ is $E_{i,j}(-\lambda)$. If f is a transvection of the space $\mathbb{F}_q^n$, then there exists a basis $\mathcal{B}$ of $\mathbb{F}_q^n$ such that $M_{\mathcal{B}}(f) = E_{i,j}(\lambda)$.

Lemma 1.2.1. $\mathrm{SL}_n(q)$ *is generated by elementary matrices.*

Proof. Multiplying a matrix on the left by an elementary matrix corresponds to adding a multiple of one row to another row. Hence, our claim is equivalent to the elementary result from linear algebra that any matrix in $\mathrm{SL}_n(q)$ can be reduced to the identity matrix by a finite sequence of elementary row operations. □

Lemma 1.2.2. *Every elementary matrix is in some conjugate of N.*

Proof. First, we suppose that $n > 2$. Since $E_{2,1}(1) \in N$, it suffices to show that any elementary matrix is conjugate in $\mathrm{SL}_n(q)$ to $E_{2,1}(1)$.

Define $T = E_{i,j}(\lambda)$. We will abuse notation and treat T as a linear map and a matrix at the same time. If $e_1, \dots, e_n$ are vectors from the canonical basis of $\mathbb{F}_q^n$, then $e_kT = e_k$ if $k \neq i$ and $e_iT = e_i + \lambda e_j$.

[1]Note that this can also be seen as the action of $\mathrm{PSL}_n(q)$ on the projective space $\mathbb{P}^{n-1}(\mathbb{F}_q)$.

We define a basis $\{f_1, \dots, f_n\}$ of $\mathbb{F}_q^n$ by taking $f_1 = \lambda e_j$, $f_2 = e_i$ and $f_k = e_m$ for $k \geqslant 3$ and $m \neq i, j$, ensuring that all f_k are different. Thus, $f_2 T = f_1 + f_2$ and $f_k T = f_k$ for $k \neq 2$. Hence, the matrix representation of T relative to the basis $\{f_1, \dots, f_n\}$ is $E_{2,1}(1)$. Thus, if A is the change-of-basis matrix from $\{e_1, \dots, e_n\}$ to $\{f_1, \dots, f_n\}$, $E_{i,j}(\lambda) = A^{-1} E_{2,1}(1) A$. At this point we have shown that $E_{i,j}(\lambda)$ and $E_{2,1}(1)$ are conjugate in $\mathrm{GL}_n(q)$.

For an arbitrary $c \in \mathbb{F}_q^\times$ we define the matrix A_c as $e_k A_c = f_k$ if $k < n$ and $e_n A_c = c f_n$. We observe that $T = E_{i,j}(\lambda) = A_c^{-1} E_{2,1}(1) A_c$. The reason of this is that both sides of the equation are equal at $f_1, \dots, f_{n-1}, c f_n$ and that $n > 2$. The rows of A_c are the same as the rows of A except for the last, which is c times the last row of A. As $\det A_c = c \det A$, by taking $c = (\det A)^{-1}$, the result follows.

When $n = 2$ any elementary matrix is of the form $\left(\begin{smallmatrix} 1 & \lambda \\ 0 & 1 \end{smallmatrix}\right)$ or $\left(\begin{smallmatrix} 1 & 0 \\ \lambda & 1 \end{smallmatrix}\right) \in N$, where $\lambda \in \mathbb{F}_q^\times$. Since $\left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right) \in \mathrm{SL}_2(q)$, the following establishes the result:

$$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}^{-1} \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -\lambda & 1 \end{pmatrix}. \tag{1.1}$$

□

Lemma 1.2.3. *If $(n, q) \neq (2, 2), (2, 3)$, then $\mathrm{SL}_n(q)$ is perfect.*

Proof. We first consider the case $n > 2$. Recall that the conjugate of a commutator is a commutator. Therefore, due to the proof of Lemma 1.2.2 it suffices to show that $E_{2,1}(1)$ is a commutator. Since $n \geqslant 3$,

$$E_{2,1}(1) = E_{2,3}(1) E_{3,1}(1) E_{2,3}(-1) E_{3,1}(-1) = [E_{2,3}(-1), E_{3,1}(-1)]. \tag{1.2}$$

If $n = 2$ and $q > 3$, then there exists $x \in \mathbb{F}_q^\times$ with $x^2 \neq 1$. From the proof of Lemma 1.2.2 it suffices to show that $\left(\begin{smallmatrix} 1 & 0 \\ \lambda & 1 \end{smallmatrix}\right)$ is a commutator. Take $y \in \mathbb{F}_q$ arbitrary,

$$\begin{pmatrix} 1 & 0 \\ y(x^2 - 1) & 1 \end{pmatrix} = \left[\begin{pmatrix} 1 & 0 \\ y & 1 \end{pmatrix}, \begin{pmatrix} x & 0 \\ 0 & x^{-1} \end{pmatrix} \right]$$

whence we get the desired result. □

Finally, we are ready to prove the simplicity of $\mathrm{PSL}_n(q)$ if $(n, q) \neq (2, 2), (2, 3)$. In the two cases in which $\mathrm{PSL}_n(q)$ is not simple we have the isomorphisms $\mathrm{PSL}_2(2) \cong S_3$ and $\mathrm{PSL}_2(3) \cong A_4$ [Wil09, p. 44].

Theorem 1.2.4. *If $(n, q) \neq (2, 2), (2, 3)$, then $\mathrm{PSL}_n(q)$ is simple.*

Proof. Define $Z = Z(\mathrm{SL}_n(q))$. We know that $\mathrm{PSL}_n(q)$ acts faithfully and primitively on Ω. Recall that if $\omega = \langle (1, 0, \dots, 0) \rangle$, the group $\mathrm{Stab}_{\mathrm{SL}_n(q)}(\omega)$ contains the normal abelian subgroup N. Thus, as

$$\mathrm{Stab}_{\mathrm{SL}_n(q)}(\omega) Z / Z = \mathrm{Stab}_{\mathrm{PSL}_n(q)}(\omega)$$

we have that $\mathrm{Stab}_{\mathrm{PSL}_n(q)}(\omega)$ contains the normal abelian subgroup $A = NZ/Z$. By combining Lemma 1.2.1 and Lemma 1.2.2 we conclude that $\mathrm{SL}_n(q) = \langle N \rangle^{\mathrm{SL}_n(q)}$, and therefore $\mathrm{PSL}_n(q) = \langle A \rangle^{\mathrm{PSL}_n(q)}$. Finally, by Lemma 1.2.3 $\mathrm{PSL}_n(q)$ is perfect except when $(n, q) = (2, 2)$ or $(2, 3)$, and applying Iwasawa's Lemma 1.1.1, the result follows. $\square$

1.3 Bilinear forms

The rest of the classical simple groups are defined as the isometry groups of forms on a vector space. In this section we will define different types of bilinear forms and conjugate-symmetric sesquilinear forms, and classify them. Each vector space V in this section is assumed to be finite dimensional.

1.3.1 Definitions

A *bilinear form* on a k-vector space V is a map $f\colon V \times V \to k$ that is linear in each argument. The pair (V, f) is said to be a *bilinear space*. We say that the form is *symmetric* if $f(u, v) = f(v, u)$ and *alternating* if $f(v, v) = 0$.

Now we present the conjugate-symmetric sesquilinear[2] forms. For these forms to make sense we consider fields of order q^2 (q is a power of a prime), and we write $\overline{x} = x^q$ for every $x \in \mathbb{F}_{q^2}$[3]. Hence, a *conjugate-symmetric sesquilinear form* over a $\mathbb{F}_{q^2}$-vector space V is a map $f\colon V \times V \to \mathbb{F}_{q^2}$ that satisfies $f(\lambda u + v, w) = \lambda f(u, w) + f(v, w)$ and $f(w, v) = \overline{f(v, w)}$ for all vectors v, w and scalars λ. The pair (V, f) is said to be a *sesquilinear space*.

We say that u and v are *perpendicular* if $f(u, v) = 0$, and we denote this by $u \perp v$. For any $S \subseteq V$, we write $S^{\perp} = \{v \in V \mid v \perp s \text{ for all } s \in S\}$, and call this subset the *perpendicular space* of S. The *radical* of f, $\operatorname{rad} f$, is the perpendicular space of V. We say that f is *non-singular* if $\operatorname{rad} f = 0$, and *singular* otherwise.

Let $\mathcal{B} = \{e_1, \dots, e_n\}$ be a basis of V. The matrix A whose (i, j) component is $f(e_i, e_j)$ is called the matrix of f with respect to $\mathcal{B}$, and it is denoted as $M_{\mathcal{B}}(f)$. It is easy to show that f is singular if and only if the determinant of the associated matrix is zero.

Given any $W \leqslant V$, $(W, f_{|W})$ is a bilinear/sesquilinear space. Of course, if f is (symmetric or alternating), so is $f_{|W}$. In general, non-singularity is not inherited. When $f_{|W}$ is non-singular, we say that the subspace W is *non-singular*. In this case, it can be shown that $V = W \oplus W^{\perp}$ and that $W^{\perp}$ is also non-singular [Wil09, p. 56].

Suppose that f is a bilinear/sesquilinear form on vector space V. An *isometry* relative to f is a k-isomorphism $g\colon V \to V$ that satisfies $f(ug, vg) =$

[2]From the Latin numerical prefix *sesqui-* which means "one and a half" because the form is linear on the first component plus half-linear (semi-linear) on the second one.

[3]The map given by such assignment is called the Frobenius automorphism of $\mathbb{F}_{q^2}$.

$f(u, v)$ for all $u, v \in V$. Two forms on V are said to be *equivalent* if they become equal after a change of basis.

1.3.2 Classification of alternating forms

Before classifying the alternating forms, we will see that there are no spaces of odd dimension equipped with an alternating bilinear form.

Theorem 1.3.1. *Let (V, f) be a non-singular alternating bilinear space. Then $\dim V$ is even.*

Proof. By induction on $\dim V$. If $\dim V = 1$ with basis $\{v\}$, then since f is an alternating form $f(\lambda v, \mu v) = \lambda\mu f(v, v) = 0$. This contradicts the hypothesis of non-singularity.

If $\dim V = 2$ we get the desired result, so suppose that $\dim V \geqslant 3$. Take $v \in V - \{0\}$. Since f is non-singular, there exists $w \in V$ such that $f(v, w) = 1$ and that $W = \langle v, w\rangle$ is of dimension 2. The matrix of $f_{|W}$ with respect to the basis $\{v, w\}$ is $\left(\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right)$, which is invertible. Hence, the restriction $f_{|W}$ is non-singular. Therefore, $V = W \oplus W^{\perp}$ and $W^{\perp}$ is also non-singular. By induction hypothesis $\dim W^{\perp}$ is even and thus $\dim V$ so is. □

Let (V, f) be a non-singular alternating bilinear space. In the proof of Theorem 1.3.1 we have seen that for every non-zero vector $v \in V$ we can find $w \in V$ such that: $f(v, w) = 1$; $V = W \oplus W^{\perp}$, where $W = \langle v, w\rangle$ is of dimension 2; and both $f_{|W}$ and $f_{|W^{\perp}}$ are non-singular. The pair (v, w) is called a *hyperbolic pair*. By induction one can easily show that there is a basis $\{e_1, \dots, e_m, f_1, \dots, f_m\}$ of V with $u \perp v = 0$ for all basis vectors u, v except $f(e_i, f_i) = -f(f_i, e_i) = 1$. A basis that fulfills such conditions is called a *symplectic basis* and f is said to be a *symplectic form*. The matrix of f with respect to $\{e_1, \dots, f_m\}$ is $\left(\begin{smallmatrix} 0 & I_m \\ -I_m & 0 \end{smallmatrix}\right)$. Therefore, up to equivalence, there is only one alternating form on a non-singular alternating bilinear space.

1.3.3 Classification of conjugate-symmetric sesquilinear forms

As in the case of the alternating forms, up to equivalence, there is only one conjugate-symmetric sesquilinear form. In fact, we will prove that the matrix of the conjugate-symmetric sesquilinear form with respect to some basis is the identity matrix.

Theorem 1.3.2. *Let f be a non-singular conjugate-symmetric sesquilinear form on a $\mathbb{F}_{q^2}$-vector space V. Then there is a basis of V of mutually perpendicular vectors each of norm 1, i.e. $f(v, v) = 1$ for all basis vectors v. Such a basis is called an* orthonormal basis.

Proof. By induction on $\dim V$. For the case $V = \{0\}$ we use the empty basis and the result follows. Now let $\dim V = n \geqslant 1$. We claim that there is a vector v with $f(v,v) \neq 0$. Otherwise, for any u, w vectors we get that

$$\begin{aligned} 0 &= f(u+\lambda w, u+\lambda w) \\ &= f(u,u) + \overline{\lambda} f(u,w) + \lambda f(w,u) + \lambda\overline{\lambda} f(w,w) \\ &= \overline{\lambda} f(u,w) + \lambda f(w,u). \end{aligned} \tag{1.3}$$

Now choose two values of λ that form a $\mathbb{F}_q$-basis of $\mathbb{F}_{q^2}$. For instance, $\lambda_1 = 1$ and some scalar λ_2 with $\lambda_2 \neq \overline{\lambda_2}$, that is $\lambda_2 \in \mathbb{F}_{q^2} - \mathbb{F}_q$. Solving the equation (1.3) for these values we conclude that $f \equiv 0$, which contradicts the hypothesis of non-singularity.

Let v be some vector with $f(v,v) \neq 0$. Then $f(v,v) = \overline{f(v,v)}$, and $f(v,v)$ is in the fixed field $\mathbb{F}_q$ of the field automorphism $x \mapsto x^q$. Since the multiplicative group of the field is cyclic of order $q^2 - 1 = (q+1)(q-1)$, there is $\lambda \in \mathbb{F}_{q^2}$ such that $\lambda\overline{\lambda} = \lambda^{q+1} = f(v,v)$. Therefore, define $v' = \lambda^{-1} v$, so that $f(v',v') = 1$. Now let W be the subspace spanned by v. Since $f_{|W}$ is non-singular, we get that $V = W \oplus W^{\perp}$, and by induction there is an orthonormal basis of $W^{\perp}$ $\{v_1, \ldots, v_{n-1}\}$. Clearly $\{v_1, \ldots, v_{n-1}, v'\}$ is an orthonormal basis of V. □

1.3.4 Classification of symmetric forms in odd characteristic

The classification of symmetric forms not only does require more work than the previous ones, but also it depends on the characteristic of the underlying field. After some preliminaries, we will be able to answer the equivalence question for finite fields of odd characteristic.

Assume for the rest of this section that k is a field of odd characteristic and that V a finite dimensional k-vector space.

Suppose that f is a symmetric form on V, then define $Q\colon V \to k$ as $Q(v) = f(v,v)$, and call Q the associated *quadratic form*. Observe that $Q(\lambda v) = \lambda^2 Q(v)$ for all $\lambda \in k$ and $v \in V$, and that for all $u, v \in V$

$$Q(u+v) = f(u+v, u+v) = Q(u) + 2f(u,v) + Q(v). \tag{1.4}$$

Therefore, since $\operatorname{char} k \neq 2$, we have that $f(u,v) = \frac{1}{2}(Q(u+v) - Q(u) - Q(v))$, i.e. the quadratic form Q completely determines the bilinear form f.

Theorem 1.3.3. *Let f be a symmetric bilinear form on V, then V has an orthogonal basis $\mathcal{B} = \{v_1, \ldots, v_n\}$. That is, there are some $\lambda_i \neq 0$ with $M_{\mathcal{B}}(f) = \operatorname{diag}(\lambda_1, \ldots, \lambda_r, 0, \ldots, 0)$. Furthermore, $\{v_{r+1}, \ldots, v_n\}$ is a basis of $\operatorname{rad} f$; and for every $1 \leqslant i \leqslant r$, v_i can be replaced by $c_i v_i$ for every $c_i \in k^{\times}$ and each λ_i can be any non-zero field element in the image of the restriction of Q to $\langle v_1, \ldots, v_{i-1}\rangle^{\perp}$.*

Proof. Assume that $f \neq 0$, as otherwise the theorem trivially holds in that case. The proof is by induction on $n = \dim V$. The case $n = 1$ is trivial, so suppose that $n \geqslant 2$. First note that $Q(v_1) \neq 0$ for some $v_1 \in V$. Otherwise, by (1.4) we would have $f(u, v) = 0$ for all $u, v \in V$, a contradiction since $f \neq 0$. Define $W = \langle v_1 \rangle$ so W is non-singular. Hence, $V = W \oplus W^{\perp}$. By induction, $W^{\perp}$ has an orthogonal basis $\{v_2, \dots, v_r\}$ such that $Q(v_i) = \lambda_i \neq 0$ when $2 \leqslant i \leqslant r$ and $Q(v_i) = 0$ if $r + 1 \leqslant i \leqslant n$, for some r. Hence, the equality $M_{\mathcal{B}}(f) = \operatorname{diag}(\lambda_1, \dots, \lambda_r, 0, \dots, 0)$ is clear.

If $v \in V$ then $v \in \operatorname{rad} f$ if and only if $v \perp v_i$ for every i. Suppose that $v = \sum_i a_i v_i$, therefore if $j > r$, $f(v, v_j) = 0$, and $f(v, v_j) = a_j \lambda_j$ otherwise. Hence, $v \in \operatorname{rad} f$ if and only if $a_j = 0$ for $1 \leqslant j \leqslant r$, or equivalently $v \in \langle v_{r+1}, \dots, v_n \rangle$.

Since $Q(c_i v_i) = c_i^2 \lambda_i$, every v_i can be replaced by $c_i v_i$ for every $c_i \in k^{\times}$. The other fact is immediate recalling the main proof of the theorem. □

We now define some concepts that will be useful in what follows. If f is a symmetric form on V, then $v \in V - \{0\}$ is called *isotropic* if $Q(v) = 0$; otherwise is called *anisotropic*. For technical reasons, $v = 0$ is taken to be anisotropic. If there exists an isotropic vector then f, V and Q are called *isotropic*; otherwise are called *anisotropic*. A *totally isotropic subspace* of V is a subspace on which all non-zero vectors are isotropic. The form f and the quadratic form Q are called *universal* if $Q(V) = k$, i.e. Q is a surjective map.

Proposition 1.3.4. *Every non-singular isotropic symmetric bilinear form is universal.*

Proof. Let f be any non-singular isotropic symmetric bilinear form. Take an isotropic vector $u \in V - \{0\}$. Since f is non-singular, there is some $w \in V$ with $f(u, w) = \lambda \neq 0$. We can replace w by $\frac{w}{2\lambda}$ so that $f(u, w) = 1/2$. Define $v = cu + w$, where $c \in k$ will be determined later. Then, as u is isotropic,

$$Q(v) = f(cu + w, cu + w) = 2cf(u, w) + f(w, w) = c + f(w, w).$$

Finally, for $a \in k$, set $c = a - f(w, w)$. Therefore

$$Q(v) = a - f(w, w) + f(w, w) = a.$$

□

Suppose now that $k = \mathbb{F}_q$, where q is a power of an odd prime. Let $k^{\times 2}$ denote the subgroup of squares in the multiplicative group $k^{\times}$. The map $\theta \colon k^{\times} \to k^{\times}$ given by $x \mapsto x^2$ is a group homomorphism and $\ker \theta = \{\pm 1\}$. Hence, $|k^{\times} : k^{\times 2}| = 2$ and the two cosets correspond to the squares and the non-squares in $k^{\times}$.

If $b \in k^{\times}$ is a non-square, denote by K the splitting field of the polynomial $x^2 - b$, so $[K : k] = 2$ and $|K| = q^2$. Therefore, we may write

$K = \{a + c\sqrt{b} \mid a, c \in k\}$. Moreover, as the map $x \mapsto x^q$ generates the Galois group $\mathrm{Gal}(K/k) \cong C_2$, the field norm is given by

$$\mathrm{N}_{K/k}(a + c\sqrt{b}) = \prod_{\sigma \in \mathrm{Gal}(K/k)} \sigma(a + c\sqrt{b}) = (a + c\sqrt{b})^{q+1}.$$

Note that the restriction $N := \mathrm{N}_{K/k} \colon K^\times \to k^\times$ is a homomorphism and that $\ker N = \{\alpha \in K \mid \alpha^{q+1} = 1\}$, which is the unique subgroup of order $q+1$ of $K^\times$. Thus, the image of N has order $\frac{q^2-1}{q+1} = q-1$, i.e. N is surjective.

In general, if f is a symmetric form and $\lambda \in k^\times$, then $f_\lambda(u, v) = \lambda f(u, v)$ is also a symmetric form and $Q_\lambda(v) = \lambda Q(v)$ is the corresponding quadratic form. Observe that f is universal if and only if f_λ is universal for all $\lambda \in k^\times$.

Proposition 1.3.5. *Let k be a finite field and let f be non-singular symmetric bilinear form on a k-vector space V of dimension $n \geqslant 2$. Then f is universal. Conversely, if* $\dim V = 1$ *then f is not universal.*

Proof. By Proposition 1.3.4 we can assume that f is anisotropic. Note that it suffices to show only the case $n = 2$, since every vector space (of dimension $n \geqslant 2$) has a subspace of dimension 2, and applying the proof to this particular subspace the claim will follow.

By Theorem 1.3.3 we may assume that there is an orthogonal basis $\mathcal{B} = \{v_1, v_2\}$ for which $M_{\mathcal{B}}(f) = \mathrm{diag}(\alpha, -\beta)$, where both α and β are non-zero; and by scaling the form we may assume that $\alpha = 1$. Let $v = av_1 + bv_2$ be any non-zero vector of V, then $Q(v) = a^2 - \beta b^2 \neq 0$ because f is anisotropic. Hence, β is a non-square in k, so we consider $K = k(\sqrt{\beta})$. From the definition of the field norm $\mathrm{N}_{K/k}$ we have that $\mathrm{N}_{K/k}(a + b\sqrt{\beta}) = a^2 - \beta b^2$ and thus $\mathrm{N}_{K/k}(a + b\sqrt{\beta}) = Q(v)$. Finally, by the discussion before this theorem we know that $\mathrm{N}_{K/k} \colon K^\times \to k^\times$ is surjective and consequently Q is surjective.

Suppose now that $\dim V = 1$. Recall that $Q(\lambda v) = \lambda^2 Q(v)$ for all $\lambda \in k$ and $v \in V$. Therefore, one of the following must occur: $Q(V) \subseteq k^{\times 2} \subsetneq k$ or $Q(V) \subseteq c \cdot k^{\times 2} \subsetneq k$, where c is a non-square of k. $\square$

Theorem 1.3.6. *Let k be a field of odd order and let f be a non-singular symmetric bilinear form on a k-vector space V of dimension $n \geqslant 2$. Then there is a basis of V which the matrix with respect to f equals* $\mathrm{diag}(1, \dots, 1, d)$, *for some $d \in k^\times$.*

Proof. By Theorem 1.3.3 and Proposition 1.3.5 we know that there is some $v_1 \in V$ with $Q(v_1) = 1$. By these same results we can continue choosing vectors v_i in an orthogonal basis, where $Q(v_i) = 1$, as long as $\langle v_1, \dots, v_{i-1}\rangle^\perp$ has dimension greater or equal than 2. This can be done for all $1 \leqslant i \leqslant n-1$. When $i = n$ universality is no longer guaranteed, so v_n is chosen as a vector from $\langle v_1, \dots, v_{n-1}\rangle^\perp$ (which may have dimension 1, so we cannot apply Proposition 1.3.5) with $Q(v_n) = d \neq 0$, as in the proof of Theorem 1.3.3. $\square$

Therefore, there are exactly two equivalence classes of non-singular symmetric bilinear forms over fields of odd order: the ones which have a matrix as in the theorem above with $d = 1$, and those which have such a matrix with $d = a$, where a is some non-square. Note that these two types are not equivalent, one has determinant 1, which lies in the coset of squares and the other has determinant a, which lies in the other coset.

Before defining forms of *plus type* and *minus type*, we will give an example to motivate these definitions. Suppose that $\dim V = 2$. Let f_1 and f_2 be two symmetric forms given with respect to an orthogonal basis $\{x, y\}$ by $f_1(x, x) = f_1(y, y)$, and $f_2(x, x) = 1, f_2(y, y) = d$ for some non-square scalar d. If -1 is a square in k (write $-1 = i^2$), then $f_1(x + iy, x + iy) = 0$ and $f_2(x + \lambda y, x + \lambda y) = 1 + \lambda^2 d$, which cannot be 0 (otherwise, $d = -\lambda^{-2} = (\lambda^{-1} i)^2$, which is impossible). Alternatively, when -1 is not a square, $-d$ is a square (write $-d = \lambda^{-2}$ for some $\lambda \in k^\times$), so $f_2(x + \lambda y, x + \lambda y) = 0$ and $f_1(x + \lambda y, x + \lambda y) \neq 0$. It is well known that -1 is a square in $\mathbb{F}_q$ if and only if $q \equiv_4 1$. Hence, there is a non-zero isotropic vector for f_1 if and only if $q \equiv_4 1$. Moreover, there is a non-zero isotropic vector for f_2 if and only if $q \equiv_4 3$. Therefore, we say that a form is of *plus type* if there is an isotropic vector; otherwise we say that is of *minus type*.

In general, a form in $2m$ dimensions is said to be of *plus type* if there exists a totally isotropic subspace of dimension m; otherwise is called of *minus type*. The *Witt index* of the form is defined as the maximal dimension of a totally isotropic subspace. It can be shown that when the form is of plus type the Witt index equals m and the forms with minus type have Witt index $m - 1$ [Wil09, p. 59].

1.4 Symplectic groups

The *symplectic group*[4] $\mathrm{Sp}(V, f)$ is defined as the isometry group of a symplectic form (non-singular alternating) f on a vector space $V \cong \mathbb{F}_q^{2m}$ (when the form f is clear from the context, we simply write $\mathrm{Sp}(V)$). In other words, $\mathrm{Sp}(V, f)$ is the subgroup of $\mathrm{GL}(V)$ that consists of elements g such that $f(ug, vg) = f(u, v)$ for every $u, v \in V$. If h is another symplectic form on V, by the classification of alternating forms, h and f are equivalent. Hence, the resulting symplectic group respect to h is conjugate to $\mathrm{Sp}(V, f)$ in $\mathrm{GL}(V)$. Equivalently, relative to appropriately chosen bases, $\mathrm{Sp}(V, f)$ and $\mathrm{Sp}(V, h)$ are isomorphic to a subgroup of $\mathrm{GL}_{2m}(q)$. This resulting group of matrices is denoted by $\mathrm{Sp}_{2m}(q)$, and by the classification of alternating forms it can

[4]This term is due to H. Weyl, who replaced the previous confusing name "complex group", as these groups have nothing to do with complex numbers. As a curiosity, the term "symplectic" comes from the Greek word *symplektikos*, which is the Greek cognate of "complex".

be easily shown that

$$\mathrm{Sp}_{2m}(q) = \left\{ X \in \mathrm{GL}_{2m}(q) \mid X^T \left(\begin{smallmatrix} 0 & I_m \\ -I_m & 0 \end{smallmatrix}\right) X = \left(\begin{smallmatrix} 0 & I_m \\ -I_m & 0 \end{smallmatrix}\right) \right\}. \tag{1.5}$$

The centre Z of $\mathrm{Sp}_{2m}(q)$ consists of the matrices I_{2m} and $-I_{2m}$ if $\operatorname{char} \mathbb{F}_q \neq 2$, and $Z = \{I_{2m}\}$ if $\operatorname{char} \mathbb{F}_q = 2$. The quotient group $\mathrm{Sp}_{2m}(q)/Z$ is called the *projective symplectic group* and is denoted as $\mathrm{PSp}_{2m}(q)$. In Exercise A.1.2 we compute the orders of these groups.

We start by considering the symplectic groups when $m = 1$. By (1.5), a simple computation yields that $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \in \mathrm{Sp}_2(q)$ if and only if $ad - bc = 1$. Thus, $\mathrm{Sp}_2(q) = \mathrm{SL}_2(q)$.

From now on in this section, V will be a $\mathbb{F}_q$-vector space of dimension $2m$ and f will be symplectic form on V. After proving some technical results we will be ready to apply Iwasawa's lemma to $\mathrm{PSp}_{2m}(q)$.

Lemma 1.4.1. *Let $v \in V$ and $\varphi \in V^*$. A transvection $T_v(\varphi)$ is in $\mathrm{Sp}(V, f)$ if and only if $\ker \varphi = v^\perp$. In that case there is $\lambda \in \mathbb{F}_q$ such that $xT_v(\varphi) = x + \lambda f(x, v)v$. $T_v(\varphi)$ is called a* symplectic transvection *and we write $T_v(\lambda)$ instead of $T_v(\varphi)$.*

Proof. $\tau = T_v(\varphi)$ is symplectic if and only if

$$f(x, y) = f(x\tau, y\tau) = f(x, y) + y\varphi f(x, v) + x\varphi f(v, y) + f(v, v).$$

The equation above is equivalent to

$$y\varphi f(x, v) + x\varphi f(v, y) = 0. \tag{1.6}$$

Suppose this holds and let $w \in V$ be such that $f(v, w) \neq 0$, which exists since f is non-singular. Then $w\varphi f(x, v) + x\varphi f(v, w) = 0$, and hence $x\varphi = -\frac{w\varphi}{f(v,w)} f(x, v) = \lambda f(x, v)$ and $\lambda \in \mathbb{F}_q$ is independent of x. It is clear that in this case $\ker \varphi = v^\perp$. For the converse, if $\ker \varphi = v^\perp = \ker f(\cdot, v)$, there is λ such that $x\varphi = \lambda f(x, v)$. Clearly (1.6) holds ($f$ is an alternating form), and hence τ is symplectic. □

Proposition 1.4.2. *Let $T_v(\lambda)$ and $T_v(\mu)$ be symplectic transvections. Then:*

(i) $T_v(\lambda)T_v(\mu) = T_v(\lambda + \mu)$,

(ii) $T_{\mu v}(\lambda) = T_v(\lambda\mu^2)$,

(iii) $T_v(\lambda)^{-1} = T_v(-\lambda)$,

(iv) $T_v(\lambda)^g = T_{vg}(\lambda)$ *for any* $g \in \mathrm{Sp}(V)$.

Proof. Easy computation. □

Lemma 1.4.3. *The subgroup S generated by all symplectic transvections equals $\mathrm{Sp}(V)$.*

Proof. Part 1: S acts transitively on $V^2 - \{(0,0)\}$.

Let v, w be two distinct non-zero vectors. If $f(v,w) = \lambda \neq 0$, then $vT_{v-w}(\lambda^{-1}) = w$. Otherwise, $f(v,w) = 0$. We aim to find an x such that $f(v,x) \neq 0 \neq f(w,x)$. Such x exists because if not, there exist y, z such that $f(v,y) = 0 = f(w,z)$ and $f(v,z) \neq 0 \neq f(w,y)$. With this in mind, we see that a suitable linear combination of y and z has the required properties. Now, we can construct T_1 and T_2 symplectic transvections such that $vT_1 = z$ and $zT_2 = w$. Hence, we can map v to z and z to w.

Part 2: S acts transitively on the set of hyperbolic pairs.

We have to show that there is a product of transvections which maps $\{e_i, f_i\}$ to $\{e_j, f_j\}$. By Part 1, there is a transvection T_1 mapping e_i to e_j. If we find a transvection T_2 such that $f_iT_1T_2 = f_j$ and $e_jT_2 = e_j$, then the claim is proven. We divide into two cases.

If $f(f_iT_1, f_j) = \lambda \neq 0$, by Part 1, defining $T_2 = T_{f_iT_1 - f_j}(\lambda^{-1})$ we get $f_iT_1T_2 = f_j$ and since $f(e_j, f_iT_1 - f_j) = f(e_i, f_i) - f(e_j, f_j) = 0$, $e_jT_2 = e_j$, as we wanted.

Now suppose that $f(f_iT_1, f_j) = 0$. In this case $f(f_iT_1, e_j + f_iT_1) = -1$. Moreover, $f(e_j + f_1T_1, f_j) = f(e_j, f_j) = \lambda \neq 0$. Furthermore, $f(e_j, -e_j) = 0 = f(e_j, e_j + f_iT_1 - f_j)$. Thus, by Part 1 there exist $T_{2,1} = T_{-e_j}(-1)$ and $T_{2,2} = T_{e_j + f_1T_1 - f_j}(\lambda^{-1})$ that both fix e_j, and that $f_1T_1T_{2,1} = e_j + f_iT_1$ and $(e_j + f_1T_1)T_{2,2} = f_j$. Therefore, defining T_2 as the composition $T_{2,2}$ of $T_{2,1}$ gives the desired result.

Now we are ready to prove that $S = \mathrm{Sp}(V)$. We prove this by induction on $\dim V$. When $\dim V = 2$ by Lemma 1.2.1 the result follows since $\mathrm{SL}_2(q) = \mathrm{Sp}_2(q)$.

Suppose now that $\dim V > 2$. Let (v,w) be a hyperbolic pair and define $W = \langle v, w\rangle$. Since for any $g \in \mathrm{Sp}(V)$ (vg, wg) is a hyperbolic pair, and by Part 2 there is some $T \in S$ with $vgT = v$ and $wgT = w$, then $g_{|W}T = 1_W$. Furthermore, as $g_{|W^\perp}T \in \mathrm{Sp}(W^\perp)$ by induction hypothesis $g_{|W^\perp}T$ is a product of symplectic transvections $t_1, \ldots, t_r$ on $W^\perp$. Since $V = W \oplus W^\perp$, each t_i can be extended to a symplectic transvection on V, t_i'. This extension is done in the following way. By Lemma 1.4.1 we have that $xt_i = x + \lambda f(x,v)v$ for some scalar λ and $v \in W^\perp$, for every $x \in W^\perp$. Hence, $xt_i' = xt_i$ for all $x \in V$ defines a transvection t_i' in $\mathrm{Sp}(V)$ and the equality $gT = t_1' \ldots t_r'$ holds. Finally, it follows that $g = t_1' \ldots t_r' T^{-1} \in S$, as desired. □

Corollary 1.4.4. $\mathrm{Sp}_{2m}(q) \leqslant \mathrm{SL}_{2m}(q)$.

Proof. Any transvection has determinant 1. □

Lemma 1.4.5. *$\mathrm{Sp}_{2m}(q)$ is perfect except when $(m,q) = (1,2)$, $(1,3)$ and $(2,2)$*

Proof. By Lemma 1.4.3 it suffices to show that every symplectic transvection is a commutator. We prove this by induction on $\dim V$ ($V \cong \mathbb{F}_q^{2m}$ as always).

We start with the inductive step, when $\dim V \geqslant 4$. Since the conjugate of a commutator is a commutator again, $\mathrm{Sp}(V)$ acts on $V - \{0\}$ transitively and by property iv) from Proposition 1.4.2, it suffices to verify that the transvections $T_v(\lambda)$, for a fixed vector v, are a commutator for every scalar λ. Let $\mathcal{B} = \{e_1, \ldots, f_m\}$ be a symplectic basis of V and $T = T_{e_1}(\lambda)$ be a symplectic transvection. Clearly $W = \langle e_2, f_2 \rangle \leqslant e_1{}^{\perp}$. The restriction $\tau = T_{|W^{\perp}}$ belongs to $\mathrm{Sp}(W^{\perp})$, and by induction it is a commutator. Since $W \leqslant e_1{}^{\perp}$, we have that $T_{|W} = 1_W$, and therefore T and the extension of τ to V coincide, as in the proof of Lemma 1.4.3. Hence, T is clearly a commutator on $\mathrm{Sp}(V)$.

Since $\mathrm{Sp}_2(q) = \mathrm{SL}_2(q)$ and since in $\mathrm{SL}_2(q)$ all transvections are commutators when $q \geqslant 4$ (see Lemma 1.2.3 from the previous section), the base of the induction is proven. When $q = 2$ induction starts at $m = 3$, and for $q = 3$ at $m = 2$. So we need to check that in both $\mathrm{Sp}_4(3)$ and $\mathrm{Sp}_6(2)$ every symplectic transvection is a commutator. We prove it in Exercise A.1.3. $\square$

By Corollary 1.4.4 and recalling Section 1.2 there is an action of $\mathrm{Sp}_{2m}(q)$ on the set Ω of 1-dimensional subspaces of $\mathbb{F}_q^n$. This action is transitive by Part 1 of the proof of Lemma 1.4.3. Since the kernel of this action coincides with $Z(\mathrm{Sp}_{2m}(q))$, $\mathrm{PSp}_{2m}(q)$ acts faithfully and transitively on Ω.

Lemma 1.4.6. $\mathrm{Sp}(V)$ *acts primitively on* Ω.

Proof. Since $\mathrm{SL}_2(q) = \mathrm{Sp}_2(q)$, we may assume that $\dim V > 1$. Let $B \subseteq \Omega$ with $|B| > 1$ and either $Bg = B$ or $Bg \cap B = \emptyset$ for each $g \in \mathrm{Sp}(V)$. We need to show that $B = \Omega$.

We claim that there are $\langle u \rangle, \langle v \rangle \in B$ such that $f(u, v) \neq 0$. Suppose to the contrary that $f(u, v) = 0$ for all $\langle u \rangle, \langle v \rangle \in B$. Choose $\langle u \rangle \neq \langle v \rangle$ in B. Since f is non-singular, there is $x \in V$ with $f(u, x) = 1$. Hence, (u, x) is a hyperbolic pair and define the plane $W = \langle u, x \rangle$. Set $H = \{g \in \mathrm{Sp}(V) \mid g_{|W} = 1_W\}$. It can be easily shown that $H \leqslant \mathrm{Sp}(V)$. Since $V = W \oplus W^{\perp}$, any $g \in \mathrm{Sp}(W^{\perp})$ extends to $g' \in \mathrm{Sp}(V)$ with $g'_{|W} = 1_W$, and hence $\mathrm{Sp}(W^{\perp}) = \{g_{|W^{\perp}} \mid g \in H\}$. Choose $w \in W^{\perp} - \{0\}$. Since $v \in W^{\perp}$, and $\mathrm{Sp}(V)$ acts transitively on $V - \{0\}$, there is $g \in H$ with $vg = w$. As $ug = u$, we have that $\langle u \rangle \in Bg \cap B$, so $Bg = B$. Moreover, $\langle w \rangle = \langle v \rangle g \in B$, and recall that w is an arbitrary non-zero vector in $W^{\perp}$. Since $m > 1$, $W^{\perp} \neq \{0\}$ and there is a hyperbolic pair (y, z) in $W^{\perp}$. Hence, $\langle y \rangle, \langle z \rangle \in B$, but $f(y, z) = 1$, which is a contradiction.

Therefore, we choose $\langle u \rangle, \langle v \rangle \in B$ with $f(u, v) = 1$. Hence, (u, v) is a hyperbolic pair. Now, take any $\langle w \rangle \in \Omega$. If $f(u, w) \neq 0$ we can suppose that (u, w) is a hyperbolic pair. By Part 2 of the proof of Lemma 1.4.3, there is $g \in \mathrm{Sp}(V)$ with $ug = u$ and $vg = w$. Since $\langle u \rangle \in Bg \cap B$, then $Bg = B$ and $\langle w \rangle \in B$. Otherwise, if $f(u, w) = 0$ there is $x \in V$ such that $f(u, x) = f(w, x) = 1$. Using the same arguments as in the previous paragraph, $\langle x\rangle \in B$, and there is also some $g \in \mathrm{Sp}(V)$ with $ug = w$ and $xg = x$. Once more, $B = Bg$ and $\langle w\rangle = \langle u\rangle g \in B$. Hence, $B = \Omega$. □

At this point we are ready to prove the simplicity of $\mathrm{PSp}_{2m}(q)$ if $(n, q) \neq (1, 2), (1, 3), (2, 2)$. The three cases in which $\mathrm{PSp}_{2m}(q)$ is not simple we have the isomorphisms $\mathrm{PSp}_2(2) = \mathrm{PSL}_2(2) \cong S_3$, $\mathrm{PSp}_2(3) = \mathrm{PSL}_2(3) \cong A_4$ and $\mathrm{Sp}_4(2) \cong S_6$ [Wil09, p. 61].

Theorem 1.4.7. *The group* $\mathrm{PSp}_{2m}(q)$ *is simple except when* $(m, q) = (1, 2)$*,* $(1, 3)$ *and* $(2, 2)$*.*

Proof. $\mathrm{PSp}_{2m}(q)$ acts faithfully and primitively (by Lemma 1.4.6) on Ω. If $\langle v\rangle \in \Omega$, set $H = \mathrm{Stab}_{\mathrm{Sp}_{2m}(q)}(\langle v\rangle)$. Define $N = \{T_v(\lambda) \mid \lambda \in \mathbb{F}_q\}$. By Proposition 1.4.2, $N \trianglelefteq H$ and $N \cong \mathbb{F}_q$, so N is abelian. For any $g \in \mathrm{Sp}_{2m}(q)$, $N^g = \{T_{vg}(\lambda) \mid \lambda \in \mathbb{F}_q\}$. Since $\mathrm{Sp}_{2m}(q)$ acts transitively on $V - \{0\}$, then $\{N^g \mid g \in \mathrm{Sp}_{2m}(q)\}$ contains all the symplectic transvections. Therefore, by Lemma 1.4.3, $\mathrm{Sp}_{2m}(q) = \langle N\rangle^{\mathrm{Sp}_{2m}(q)}$. Finally, with the exceptions noted, $\mathrm{PSp}_{2m}(q)$ is perfect by Lemma 1.4.5. Apply Iwasawa's Lemma 1.1.1. □

1.5 Unitary groups

The (*general*) *unitary group* $\mathrm{GU}_n(q)$ is defined as the isometry group of a non-singular conjugate-symmetric sesquilinear form f on a $\mathbb{F}_q^2$-vector space of dimension n. That is, $\mathrm{GU}_n(q)$ is the subgroup of $\mathrm{GL}_n(q^2)$ consisting of matrices which preserve the form f. Therefore, by the classification of conjugate-symmetric sesquilinear forms it can be easily shown that

$$\mathrm{GU}_n(q) = \left\{X \in \mathrm{GL}_n(q^2) \mid \overline{X}^T = X^{-1}\right\}$$

where $\overline{X} = (\overline{x_{i,j}})$ if $X = (x_{i,j})$. Furthermore, one can show that

$$|\,\mathrm{GU}_n(q)| = q^{n(n-1)/2}\prod_{i=1}^{n}(q^i - (-1)^i).$$

The subgroup of $\mathrm{GU}_n(q)$ which consists of matrices of determinant 1 is called *special unitary group* and is denoted by $\mathrm{SU}_n(q)$. If Z is the subgroup of scalar matrices, then $\mathrm{PSU}_n(q) = \mathrm{SU}_n(q)/Z$ is called the *projective special unitary group* and most of them are simple. As in the case for the symplectic groups, we have the following isomorphism: $\mathrm{SU}_2(q) \cong \mathrm{SL}_2(q)$. The groups which are not simple are the ones given by the isomorphism $\mathrm{PSU}_2(q) \cong \mathrm{PSL}_2(q)$ and $\mathrm{PSU}_3(2)$, which is a soluble group of order 72. We check this in Exercise A.1.4 using the GAP system [GAP22].

We simply sketch the proof of the simplicity of the unitary groups. As $\mathrm{SU}_2(q) \cong \mathrm{SL}_2(q)$, we suppose that $n > 2$. As always, we aim to apply Iwasawa's Lemma 1.1.1. We consider transvections of the same form as the symplectic transvections, i.e. $xT_v(\lambda) = x + \lambda f(x, v)$, for some isotropic vector v. By computation one can show that $T_v(\lambda) \in \mathrm{SU}_n(q)$ if and only if $\lambda^{q-1} = -1$, and in that case $T_v(\lambda)$ is called a *unitary transvection*. One can show that $\mathrm{SU}_n(q)$ is generated by unitary transvections except when $(n, q) = (3, 2)$. Now, we consider the action of $\mathrm{SU}_n(q)$ on the isotropic 1-dimensional spaces, i.e. the set of all $\langle u\rangle$ with $u \in \mathbb{F}_{q^2}^n - \{0\}$ and $f(u, u) = 0$. Since properties similar to Proposition 1.4.2 hold for unitary transvections, the set of unitary transvections for a fixed isotropic vector v form an abelian normal subgroup of the stabilizer of $\langle v\rangle$. It can be easily shown that $\mathrm{SU}_n(q)$ acts primitively on the set of isotropic 1-dimensional spaces. When $n > 3$, or $n = 3$ and $q > 2$, explicit computations show that all unitary transvections are commutators of matrices of $\mathrm{SU}_n(q)$. Finally, as $\mathrm{PSU}_n(q)$ acts faithfully on the set of isotropic 1-dimensional spaces, we get the following theorem:

Theorem 1.5.1. *The group* $\mathrm{PSU}_n(q)$ *is simple except when* $(n, q) = (2, 2), (2, 3)$ *and* $(3, 2)$.

1.6 Orthogonal groups in odd characteristic

In this section we will omit all the proofs. The interested reader on them is invited to read [Gro01], but it should be careful as we use a different notation.

In Section 1.3.4 we proved that, up to equivalence, there are exactly two non-singular symmetric bilinear forms on a $\mathbb{F}_q$-vector space V, with q odd. Let f be a non-singular symmetric bilinear form on V of dimension n. The *(general) orthogonal group* $\mathrm{GO}(V, f)$ is defined as the isometry group of f. That is, it is the subgroup of $\mathrm{GL}(V)$ that consists of the invertible linear maps g that satisfy $f(ug, vg) = f(u, v)$ for all $u, v \in V$. It is clear that for any scalar λ, $\mathrm{GO}(V, f) = \mathrm{GO}(V, \lambda f)$. As in the other classical groups, all the groups defined in this section can be regarded as some groups of matrices.

If n is even, it can be shown that f and λf are equivalent forms for any scalar λ. Alternatively, if n is odd and λ is a non-square scalar, then f and λf are never equivalent. In this case there is only one orthogonal group up to isomorphism, and we denote it as $\mathrm{GO}(V)$, or $\mathrm{GO}_n(q)$ without ambiguity (up to isomorphism). When n is even, there are two different orthogonal groups, in fact they do not even have the same order. If f is of plus type we denote $\mathrm{GO}(V, f)$ by $\mathrm{GO}^+_{2m}(q)$; and $\mathrm{GO}^-_{2m}(q)$ when the form is of minus type.

Observe that any isometry g in an orthogonal group has determinant ± 1. To see this, let J be the matrix of the form f. Then $g^T Jg = J$, and by taking the determinant map we get that $\det g = (\det g)^{-1}$. The isometries of determinant 1 form a subgroup of index 2, which is the *special orthogonal group*, and is denoted as $\mathrm{SO}_n(q)$ when n is odd, and $\mathrm{SO}^\varepsilon_n(q)$ when n is even and the corresponding form is of $\varepsilon \in \{+, -\}$ type. If n

is odd, the *projective general orthogonal group* $\mathrm{PGO}_n(q)$ and the *projective special orthogonal group* $\mathrm{PSO}_n(q)$ are the groups obtained from $\mathrm{GO}_n(q)$ and $\mathrm{SO}_n(q)$ on factoring them by their respective subgroup of scalar matrices. When $n = 2m$ is even, these groups are defined in the same way but are denoted as $\mathrm{PGO}^{\varepsilon}_{2m}(q)$ and $\mathrm{PSO}^{\varepsilon}_{2m}(q)$ if the corresponding form is of type ε.

Contrary to the other families of classical groups, in general $\mathrm{PSO}^{\varepsilon}_n(q)$[5] is not simple. In fact, it has a mysterious subgroup of index 2, which is defined as the kernel of an invariant called *spinor norm.* This invariant is a homomorphism from $\mathrm{SO}^{\varepsilon}_n(q)$ to $\mathbb{F}_q^{\times}/\mathbb{F}_q^{\times 2}$, which is defined as follows: a *reflection* orthogonal to a vector v, i.e. an isometry given by the formula

$$r_v \colon x \mapsto x - 2\frac{f(x,v)}{f(v,v)}v,$$

is mapped to the value $f(v,v)$ modulo $\mathbb{F}_q^{\times 2}$. This is extended to a well-defined and unique homomorphism on all of $\mathrm{SO}^{\varepsilon}_n(q)$. Note that for such a reflection map to exist, v must be anisotropic, so the map goes to $\mathbb{F}_q^{\times}/\mathbb{F}_q^{\times 2}$. Observe that different choices of v that are scalar multiples of each other define the same reflection map. That is why the spinor norm is defined only as a map to $\mathbb{F}_q^{\times}/\mathbb{F}_q^{\times 2}$ and not as a map to just $\mathbb{F}_q^{\times}$.

The kernel of the spinor norm is denoted as $\Omega(V,f)$. As in the other orthogonal groups, it is customary to write $\Omega_{2m+1}(q)$ and $\Omega^{\varepsilon}_{2m}(q)$. The quotients $\Omega(V,f)/\{\pm I_n\}$ are denoted as $\mathrm{P}\Omega(V,f)$; or $\mathrm{P}\Omega_{2m+1}(q)$ and $\mathrm{P}\Omega^{\varepsilon}_{2m}(q)$. Abusing of notation, we simply write $\mathrm{P}\Omega^{\varepsilon}_n(q)$ to denote $\mathrm{P}\Omega(V,f)$, where if n is odd the ε does not mean anything. We can do the same for the rest of orthogonal groups.

For the orthogonal groups we have the following simplicity theorem:

Theorem 1.6.1. *If q is odd and $n \geqslant 5$, then the group $\mathrm{P}\Omega^{\varepsilon}_n(q)$ is simple.*

Finally, we present the orders of the orthogonal groups. When the dimension of the space is odd:

$$|\,\mathrm{GO}_{2m+1}(q)| = 2q^{m^2}(q^2-1)(q^4-1)\cdots(q^{2m}-1).$$

If the space is of even dimension:

$$|\,\mathrm{GO}^{+}_{2m}(q)| = 2q^{m(m-1)}(q^2-1)(q^4-1)\cdots(q^{2m-2}-1)(q^m-1), \tag{1.7}$$

$$|\,\mathrm{GO}^{-}_{2m}(q)| = 2q^{m(m-1)}(q^2-1)(q^4-1)\cdots(q^{2m-2}-1)(q^m+1). \tag{1.8}$$

1.7 Orthogonal groups in characteristic 2

In characteristic 2 the situation is quite different. Instead of classifying the symmetric bilinear forms, in this case we classify the quadratic forms.

[5] We abuse notation and combine both notations in one symbol.

A quadratic form in $2m$ dimensions is defined to be of *plus type* if it has isotropic subspaces of dimension m, and of *minus type* if it does not. However, the groups $\mathrm{GO}_n^{\varepsilon}(q)$, $\mathrm{SO}_n^{\varepsilon}(q)$, $\mathrm{PGO}_n^{\varepsilon}(q)$ and $\mathrm{PSO}_n^{\varepsilon}(q)$ are defined in the analogous way as in odd characteristic. In characteristic 2 it can be shown that $\mathrm{GO}_{2m+1}(q) \cong \mathrm{Sp}_{2m}(q)$ and hence we omit the case of spaces of odd dimension.

As in the case of odd characteristic, in characteristic 2 there is another mysterious subgroup of index 2, which is defined to be the kernel of another invariant, the *quasideterminant*. It is a homomorphism from $\mathrm{GO}_{2m}^{\varepsilon}(q)$ to $\{\pm 1\}$. This homomorphism maps an isometry to $(-1)^k$, where k is the dimension of its fixed space. Hence, we define $\Omega_{2m}^{\varepsilon}$ as the kernel of the quasideterminant. Note that when q is odd the quasideterminant agrees with the determinant.

Theorem 1.7.1. *If q is even and $m \geqslant 3$, then $\Omega_{2m}^{\varepsilon}(q)$ is simple.*

Since we are working in characteristic 2, there is no confusion (with the notation from the previous section) in denoting the groups $\Omega_{2m}^{\varepsilon}(q)$ by $\mathrm{P}\Omega_{2m}^{\varepsilon}(q)$. The orders of the orthogonal groups in even characteristic are given by the formulae (1.7) and (1.8).

Chapter 2

Character theory

In what follows we shall assume that we are given a finite group G. Also, we may assume that $C_1, \dots, C_r$ are the distinct conjugacy classes of G with representatives $1 = g_1, \dots, g_r$, respectively. The only character theoretic prerequisites for this chapter are the first two chapters from [Isa94], which will be our main reference.

2.1 Ore's criterion

The aim of the present section is give a self-contained proof of *Ore's criterion*: in a finite group G an element g is a commutator if and only if

$$\sum_{\chi \in \operatorname{Irr} G} \frac{\chi(g)}{\chi(1)} \neq 0.$$

We start proving some technical lemmas.

Lemma 2.1.1. *Let $\mathfrak{X}$ be an irreducible complex representation of G of degree n. Suppose that $A \in M_n(\mathbb{C})$ commutes with $\mathfrak{X}(g)$ for all $g \in G$. Then A is a scalar matrix.*

Proof. Let V be the irreducible $\mathbb{C}G$-module of dimension n given by the action $v \cdot x = v\mathfrak{X}(x)$ for $v \in V$ and $x \in \mathbb{C}G$. Define the map $f\colon V \to V$ as $vf = vA$. Since A commutes with $\mathfrak{X}(g)$ for all $g \in G$, it is clear that $f \in \operatorname{End}_{\mathbb{C}G}(V)$. Finally, since V is irreducible, by Schur's Lemma we get that $\operatorname{End}_{\mathbb{C}G}(V) = \mathbb{C} \cdot 1_V$. The claim follows. $\square$

Lemma 2.1.2. *For $1 \leqslant i, j, t \leqslant r$, define the positive integer $a_{i,j,t}$ as the number of pairs $(x, y) \in C_i \times C_j$ with $g_t = xy$. Then,*

$$a_{i,j,t} = \frac{|C_i||C_j|}{|G|} \sum_{\chi \in \operatorname{Irr} G} \frac{\chi(g_i)\chi(g_j)\overline{\chi(g_t)}}{\chi(1)}.$$

Proof. Let $C_1, \dots, C_r$ be the conjugacy classes of G and $K_1, \dots, K_r$ be the corresponding class sums. It is well known that $\{K_i\}_{i=1}^r$ is a basis of $Z(\mathbb{C}G)$, and therefore $K_iK_j = \sum_{v=1}^r c_{i,j,v}K_v$. From the definition of the group algebra, the coefficient $c_{i,j,t}$ equals the coefficient of g_t in the product K_iK_j, and due to the definition of multiplication in a group algebra we conclude that

$$c_{i,j,t} = |\{(x, y) \in C_i \times C_j \mid g_t = xy\}| = a_{i,j,t}.$$

Now consider the map $f_t(x) = \sum_{\chi\in\operatorname{Irr} G} \chi(x)\overline{\chi(g_t)}$ for $x \in \mathbb{C}G$. By applying f_t to the equality $K_iK_j = \sum_{v=1}^r a_{i,j,v}K_v$, and taking $g_v \in C_v$ for all $v = 1, \dots, r$ we get

$$\sum_{\chi\in\operatorname{Irr} G} \chi(K_iK_j)\overline{\chi(g_t)} = \sum_{v=1}^r a_{i,j,v}|C_v| \sum_{\chi\in\operatorname{Irr} G} \chi(g_v)\overline{\chi(g_t)} = a_{i,j,t}|G| \tag{2.1}$$

where in the last equality we have applied the second orthogonality relation.

Fix $\chi \in \operatorname{Irr} G$ and let $\mathfrak{X}$ be any representation that affords χ. For any $z \in Z(\mathbb{C}G)$, $\mathfrak{X}(z)$ commutes with $\mathfrak{X}(g)$ for all $g \in G$, and hence, by Lemma 2.1.1, there is a unique $\lambda(z) \in \mathbb{C}$ such that $\mathfrak{X}(z) = \lambda(z)I_n$, where $n = \chi(1)$. Therefore, the map $\lambda\colon Z(\mathbb{C}G) \to \mathbb{C}$ is a $\mathbb{C}$-algebra homomorphism. If we write $\lambda_i := \lambda(K_i)$ and $\lambda_j := \lambda(K_j)$, then $\chi(K_i) = \lambda_i\chi(1)$ and $\chi(K_j) = \lambda_j\chi(1)$. Since λ is a $\mathbb{C}$-algebra homomorphism, $\chi(K_iK_j) = \lambda_i\lambda_j\chi(1)$. Thus,

$$\chi(K_iK_j) = \frac{\chi(K_i)\chi(K_j)}{\chi(1)}. \tag{2.2}$$

Combining (2.1) and (2.2) we get

$$a_{i,j,t}|G| = \sum_{\chi\in\operatorname{Irr} G} \frac{\chi(K_i)\chi(K_j)\overline{\chi(g_t)}}{\chi(1)}.$$

Finally, as characters are class functions we have that $\chi(K_i) = |C_i|\chi(g_i)$ and $\chi(K_j) = |C_j|\chi(g_j)$, where $g_i \in C_i$ and $g_j \in C_j$. The claim follows. □

Lemma 2.1.3. *Let C_i be a conjugacy class of G, $g_i \in C_i$ and $g \in G$. Then the number of pairs $(x, y) \in C_i \times C_i$ with $g = x^{-1}y$ is given by*

$$N_i(g) = \frac{|C_i|^2}{|G|} \sum_{\chi\in\operatorname{Irr} G} \frac{|\chi(g_i)|^2\chi(g)}{\chi(1)}. \tag{2.3}$$

Proof. For some t we have that $g \in C_t$. Recall the definition of $a_{i,j,t}$ from Lemma 2.1.2. It is clear that $a_{i,i,t} = |\{(x, y) \in C_i \times C_i \mid g = x^{-1}y\}|$. Moreover, if $g_i \in C_i$, then for some j we have that $g_j = g_i^{-1} \in C_j$, and it is clear that $|C_i| = |C_j|$. Apply Lemma 2.1.2 and take the complex conjugate. □

Eventually, we are ready to prove Ore's criterion.

Theorem 2.1.4 (Ore's criterion)**.** *Let G be a finite group. Then $g \in G$ is a commutator if and only if*

$$\sum_{\chi \in \operatorname{Irr} G} \frac{\chi(g)}{\chi(1)} \neq 0.$$

Proof. We define $N(g)$ to be the number of pairs $(x, y) \in G \times G$ such that $g = [x, y] = x^{-1}x^y$. For a fixed conjugacy class C_i, equation (2.3) from Lemma 2.1.3 gives us the number of pairs $(x, y) \in C_i \times C_i$ with $g = x^{-1}y$. For any $h \in C_G(y)$ the equality $g = x^{-1}y^h$ is still ensured, so there are $|C_G(y)|N_i(g) = \frac{|G|}{|C_i|}N_i(g)$ pairs $(x, y) \in C_i \times G$ with $g = [x, y]$. Finally, by summing over all conjugacy classes we get that

$$N(g) = \sum_{i=1}^{r} \frac{|G|}{|C_i|} N_i(g) = \sum_{\chi \in \operatorname{Irr} G} \frac{\chi(g)}{\chi(1)} \sum_{i=1}^{r} |C_i||\chi(g_i)|^2 = |G| \sum_{\chi \in \operatorname{Irr} G} \frac{\chi(g)}{\chi(1)}$$

where in the last equality we have applied the first orthogonality relation. The result follows. □

2.2 Computing character tables

In this section, we will present an algorithm for computing the character table of an arbitrary finite group G, which is a key problem in computational group theory. Such an algorithm was first outlined by Burnside in [Bur55, §223]. However, the one we will present in this section is an improved version of it. We mostly follow the outline from [HEO05, Ch. 7], and we fill in the details from other sources such as [Hul93], [Dix67] or [Sch90].

We will assume that the conjugacy classes of G are known or can be computed efficiently (see [CS97] for a discussion on this), so we will fix the distinct conjugacy classes $C_1, \ldots, C_r$ with representatives $1 = g_1, \ldots, g_r$, respectively. The correctness of the algorithm relies on the following result:

Proposition 2.2.1. *For $1 \leqslant j, k, l \leqslant r$, as in Lemma 2.1.2, let $a_{j,k,l}$ denote the number of pairs $(x, y) \in C_j \times C_k$ with $g_l = xy$. If $\chi_i \in \operatorname{Irr} G$, then the following holds:*

$$\frac{|C_j|\chi_i(g_j)}{\chi_i(1)} \frac{|C_k|\chi_i(g_k)}{\chi_i(1)} = \sum_{l=1}^{r} a_{j,k,l} \frac{|C_l|\chi_i(g_l)}{\chi_i(1)}. \tag{2.4}$$

Proof. Fix $\chi_i \in \operatorname{Irr} G$ and let $\mathfrak{X}_i$ be any representation that affords χ_i. Define $\lambda_i \colon Z(\mathbb{C}G) \to \mathbb{C}$ the $\mathbb{C}$-algebra homomorphism where $\mathfrak{X}_i(z) = \lambda_i(z)I_n$, $n = \chi_i(1)$, as in the proof of Lemma 2.1.2. Now let C be a conjugacy class with sum $K \in Z(\mathbb{C}G)$ and let $g \in C$. Computing the traces of the equality $\mathfrak{X}_i(K) = \lambda_i(K)I_n$ yields

$$\chi_i(1)\lambda_i(K) = \chi_i(K) = \sum_{x \in C} \chi_i(x) = |C|\chi_i(g) \tag{2.5}$$

and thus, $\lambda_i(K) = \frac{|C|\chi_i(g)}{\chi_i(1)}$. Evaluating λ_i in the equality $K_jK_k = \sum_{l=1}^r a_{j,k,l}K_l$ gives the desired result. $\square$

We denote by j' the integer such that $g_j^{-1} \in C_{j'}$. By the definition of the integer $a_{j,k,l}$ it is clear that the total number of triples $(x, y, z) \in C_j \times C_k \times C_l$ with $xy = z$ equals $|C_l|a_{j,k,l}$. Moreover, this also equals the total number of such triples with $y = x^{-1}z$, and hence $|C_l|a_{j,k,l} = |C_k|a_{j',l,k}$. Substituting this in the right-hand side of (2.4) and interchanging j and j' we get

$$\frac{|C_j|\chi_i(g_{j'})}{\chi_i(1)}\chi_i(g_k) = \sum_{l=1}^{r} \chi_i(g_l)a_{j,l,k} \tag{2.6}$$

where $1 \leqslant i, j, k \leqslant r$. Rewriting (2.6) in matrix form yields

$$\frac{|C_j|\chi_i(g_{j'})}{\chi_i(1)}(\chi_i(g_1), \ldots, \chi_i(g_r)) = (\chi_i(g_1), \ldots, \chi_i(g_r)) \begin{pmatrix} a_{j,1,1} & \cdots & a_{j,1,r} \\ \vdots & \ddots & \vdots \\ a_{j,r,1} & \cdots & a_{j,r,r} \end{pmatrix} \tag{2.7}$$

for $1 \leqslant i, j \leqslant r$. Hence, if we define M_j as the $r \times r$ matrix whose (k, l) component is $a_{j,k,l}$, what (2.7) tells us is that the r row vectors $(\chi_i(g_1), \ldots, \chi_i(g_r))$ are common eigenvectors of the matrices M_j. Such matrices are called *class matrices*.

To compute the l-th column of M_j, we determine the conjugacy classes of $y_l = x^{-1}g_l$ for all $x \in C_j$. Hence, the (k, l) component of M_j is the number of these y_l that are in C_k. If we want to work efficiently in practice, the *class map* of G —the map $f : G \to \{1, \ldots, r\}$ such that $x \in C_{f(x)}$ for every $x \in G$— should be cheap to compute[1]. Notice that the first column of M_j has a unique non-zero entry $|C_j|$ in row j', so we may assume that the first column of a class matrix is always known.

Working in a finite field

We know that the values of the character table are related to the eigenvectors of the class matrices. Computations of such eigenvectors over the complex numbers involve floating point computations. Since this is not desirable, instead of working over $\mathbb{C}$, Dixon showed in [Dix67] that the eigenvector computations could be performed in a finite field to later lift back the results to $\mathbb{C}$. We present his work in this section.

Let e be the exponent of G and let $\zeta \in \mathbb{C}$ be a primitive e-th root of unity. Recall that for every character χ of G and all $g \in G$, $\chi(g)$ is a sum of $\chi(1)$ $|g|$-th roots of unity. Thus, for all the values $\chi_i(g_j)$ of the character table we have that $\chi_i(g_j) \in \mathbb{Z}[\zeta]$.

[1]Instead of checking directly conjugacy for every representative, the number of conjugacy checks can be reduced if one compares conjugacy invariants beforehand. For example, the order of an element or the cycle decomposition in permutation groups.

Theorem 2.2.2. *Fix $\chi \in \operatorname{Irr} G$. For every $g \in G$,*

$$\frac{|\operatorname{Cl}_G(g)|\chi(g)}{\chi(1)}$$

is an algebraic integer.

Proof. It is easy to show that the abelian group $R = \langle K_1, \dots, K_r\rangle$ is in fact a ring. Let $\lambda\colon Z(\mathbb{C}G) \to \mathbb{C}$ be the homomorphism defined in the proof of Lemma 2.2.1, in this case the one that depends on the irreducible character χ. Thus, $S = \lambda(R) = \langle \lambda(K_1), \dots, \lambda(K_r)\rangle$ is a ring. It is well known that a complex number z is an algebraic integer if and only if z is contained in a subring of $\mathbb{C}$ whose additive group is finitely generated. Hence, we conclude that all the $\lambda(K_i)$ are algebraic integers. Define $K = \sum_{x\in \operatorname{Cl}_G(g)} x$. By (2.5) we have the equality, $\chi(1)\lambda(K) = |\operatorname{Cl}_G(g)|\chi(g)$, and hence the result follows. □

Corollary 2.2.3. *The equations from (2.6), and consequently (2.7), involve elements from $\mathbb{Z}[\zeta]$.*

Proof. By Theorem 2.2.2 all the numbers $|C_j|\chi_i(g_j)/\chi_i(1) \in \mathbb{Q}(\zeta)$ are algebraic integers. Moreover, it is well known that $\mathbb{Z}[\zeta]$ is the ring of integers of $\mathbb{Q}(\zeta)$, and therefore all the numbers $|C_j|\chi_i(g_j)/\chi_i(1)$ lie in $\mathbb{Z}[\zeta]$. The result follows. □

By Dirichlet's theorem [Dir37] on primes in an arithmetic progression, there exists a prime p with $e|p-1$ and $p > 2\chi_i(1)$ for $1 \leqslant i \leqslant r$. At this point the degrees of the irreducible characters are unknown to us, but by the class equation these last inequalities can be replaced with the following condition: $p > 2\lfloor\sqrt{|G|}\rfloor$. As $e|p-1$, there is an element $\omega \in \mathbb{F}_p$ with multiplicative order e. Therefore, the assignation $\zeta \mapsto \omega$ induces a ring homomorphism $\Theta\colon \mathbb{Z}[\zeta] \to \mathbb{F}_p$ in the natural way:

$$\Theta\left(\sum_{i=0}^{e-1} a_i\zeta^i\right) \longmapsto \sum_{i=0}^{e-1} \overline{a_i}\omega^i.$$

Let $X = (\chi_i(g_j))$ be the character table of G regarded as a matrix, and let Y be the $r\times r$ matrix with $Y = (|C_j|\chi_i(g_{j'}))$. By the second orthogonality relation we have that $XY = |G|I_r$, in particular $\det(X)\det(Y) \neq 0$. Hence, the rows $X_{i,:} = (\chi_i(g_1), \dots, \chi_i(g_r))$, which by (2.7) are common eigenvectors of the matrices M_j, are linearly independent. Furthermore, every prime q that divides $|G|$ also divides e, and as $e|p-1$, p does not divide $|G|$. Since $|\det(X)|$ divides $|G|$, the row vectors $\Theta(X_{i,:})$ are also linearly independent over $\mathbb{F}_p$. By Corollary 2.2.3 we can apply Θ to the equations from (2.6) to get a system of equations over $\mathbb{F}_p$. Since Θ is a ring homomorphism, these vectors are common eigenvectors of the matrices $\Theta(M_j)$.

Let $X_{i,:}$ and $X_{t,:}$ be two distinct rows of X. Note that the corresponding eigenvalues $|C_j|\chi_i(g_{j'})/\chi_i(1)$ and $|C_j|\chi_t(g_{j'})/\chi_t(1)$ are different for at least one j. Otherwise, the i-th and t-th column of the matrix Y would be the same. The same applies to any pair of distinct rows of $\Theta(X)$. Therefore, the set of rows of $\Theta(X)$ and a set of r linearly independent common eigenvectors of the matrices $\Theta(M_j)$ are the same, up to scalar multiplication in each vector.

Splitting eigenspaces of the class matrices

Now we will show a method to find a set of r linearly independent common eigenvectors of the matrices $\Theta(M_j)$. Instead of computing all the class matrices, Schneider showed in [Sch90] that it was sufficient to only compute some well-chosen columns of some class matrices.

We call a subspace of $\mathbb{F}_p^r$ a *character space* if it is spanned by some rows of $\Theta(X)$. Choose some $j > 1$ and compute M_j. For example, one could take the j that corresponds to the smallest non-trivial conjugacy class. We know that $\mathbb{F}_p^r = V_1 \oplus \cdots \oplus V_t$, where the V_i are the distinct eigenspaces of $\Theta(M_j)$. Note that all the V_i are character spaces. Furthermore, we can compute a basis in echelon form for each V_i.

If $\dim V_i = 1$ for all i, then we have found r linearly independent common eigenvectors of the matrices $\Theta(M_j)$. Otherwise, there is some V_i with $\dim V_i > 1$. In this case, we should find a different $\Theta(M_j)$ to split V_i into smaller dimensional character spaces. The next lemma tells us which class matrices will split V_i, just from looking at the first column of the class matrices, which are always known, as we noted on p. 22.

Lemma 2.2.4 ([Sch90]). *Let $\{b_1, \ldots, b_s\}$ be a basis of a character space V in echelon form. Define the subspace $W = \langle b_2, \ldots, b_s \rangle$. Then V is contained in a single eigenspace of $\Theta(M_j)$ if and only if W is fixed by $\Theta(M_j)$.*

Proof. Suppose that V is contained in a single eigenspace of $\Theta(M_j)$. Then the base vectors $b_1, \ldots, b_m$ are all eigenvectors, and hence W is fixed by $\Theta(M_j)$.

For the converse, suppose that W is fixed by $\Theta(M_j)$. Assume, for sake of contradiction, that V does not lie in a single eigenspace of $\Theta(M_j)$. Therefore, there exist $v_1, v_2 \in V$ eigenvectors for different eigenvalues λ_1 and λ_2 of $\Theta(M_j)$. As v_1 and v_2 lie in different character spaces, their first component is non-zero, so we may assume that their first entry is 1. Since b_1 is the only basis vector of V with non-zero first component (by hypothesis, the basis is in echelon form), we may assume that $v_1 = b_1 + w_1$ for some $w_1 \in W$, and in an analogous way $v_2 = b_1 + w_2$, where $w_2 \in W$. Therefore,

$$\lambda_1 v_1 = v_1\Theta(M_j) = b_1\Theta(M_j) + w_1\Theta(M_j)$$

and since by assumption W is invariant under $\Theta(M_j)$, $b_1\Theta(M_j)$ must have λ_1 as its first component. The same argument for v_2 yields $\lambda_1 = \lambda_2$, which is a contradiction. □

We conclude that a character space V (of dimension greater than 1) is contained in a single eigenspace of $\Theta(M_j)$ if and only if each of its echelonized basis vector $b_2, \dots, b_s$ is zero in position j'. Otherwise, we can use $\Theta(M_j)$ to decompose V into smaller character spaces. Note that always will exist such a j.

Let j be such that the character space V is not contained in a single eigenspace of $\Theta(M_j)$. In order to determine the action of $\Theta(M_j)$ on V it is no longer necessary to know the complete class matrix M_j, but only those columns $k_1, \dots, k_s$ where the first non-zero entry of b_i is in the k_i-th entry. We will try to use the same $\Theta(M_j)$ matrix to split more than one character space of dimension greater than 1. However, different character spaces may require the computation of different columns of M_j, in order to determine their corresponding action.

Suppose that $\mathcal{V} = \{V_1, \dots, V_t\}$ is a set of character spaces such that $\mathbb{F}_p^r = V_1 \oplus \cdots \oplus V_t$, where for at least one V_i its dimension is greater than 1. The cost of computing a class matrix M_j is proportional both to $|C_j|$ and to the number of distinct columns required to determine its action in the character spaces it splits. We quantify the computational cost of the matrix M_j in the following way: set `val` $= 0$ and for each character space $V \in \mathcal{V}$ with dimension greater than 1 and that is split by $\Theta(M_j)$, increase `val` by 1. Finally, divide `val` by $|C_j|$ and by the number of columns required to determine all the actions of the character spaces that are split by $\Theta(M_j)$. The greater the value of `val` the more cost-effective it should be to determine the class matrix M_j. So, we define the number $\texttt{BestMat}(\mathcal{V}) = j$ if M_j is the class matrix with the highest `val`.

Here we describe a procedure to compute a set of r linearly independent common eigenvectors of the matrices $\Theta(M_j)$, $1 \leqslant j \leqslant r$.

Algorithm 1: Schneider

```
1 Compute $M_j$, where $j$ fulfills $|C_j| = \min_{i>1} |C_i|$;
2 Compute the set $\mathcal{V} := \{V_1, \dots, V_t\}$ of eigenspaces of $\Theta(M_j)$ and a
    base in echelon form for each $V_i$;
3 while $\exists i : \dim V_i > 1$ do
4     $j := \texttt{BestMat}(\mathcal{V})$;
5     for $V$ not contained in a single eigenspace of $\Theta(M_j)$ do
6         Determine the action of $\Theta(M_j)$ on $V$ as described above;
7         Compute the eigenspaces $\tilde{V}_1, \dots, \tilde{V}_l \leqslant V$ of $\Theta(M_j)$ on the
            space $V$ and their respective bases in echelon form;
8         Replace $V$ by $\tilde{V}_1, \dots, \tilde{V}_l$ in $\mathcal{V}$;
9 return $\mathcal{V}$;
```

In fact, this algorithm returns a set of r one-dimensional eigenspaces, but this is not a problem at all.

Returning to the complex plane

Let $v_1, \dots, v_r$ for $\Theta(M_j)$ be r linearly independent common eigenvectors of the matrices $\Theta(M_j)$ computed using Algorithm 1. Each v_i is a multiple of some row of $\Theta(X)$, so we may assume to be of the i-th one. Thus, the first entry of v_i is a multiple of $\chi_i(1) \neq 0$, so we normalize the vector to ensure that $v_i[1] = 1$. Therefore, $v_i = \frac{1}{\chi_i(1)}\Theta(X_{i,:})$, and by the first orthogonality relation

$$\sum_{j=1}^{r} |C_j| v_i[j] v_i[j'] = \frac{|G|}{\chi_i(1)^2}.$$

All in the left-hand side of this equation is known mod p, and therefore we compute the value of $\chi_i(1)^2$ mod p. Since $1 \leqslant \chi_i(1) < p/2$ (we imposed this condition on p. 23), we can fully determine the value of the integer $\chi_i(1)$. Therefore, we can compute $\Theta(X_{i,:}) = \chi_i(1)v_i$ for $1 \leqslant i \leqslant r$, and get the matrix $\Theta(X)$.

Finally, in the following lines we carry out the reconversion to the complex field. Unfortunately, Θ is not invertible (it has non-trivial kernel), so the reconversion $\mathbb{C}$ is not immediate. However, each $\chi_i(g_j)$ is the sum of $d_i = \chi_i(1)$ powers of ζ. That is, $\chi_i(g_j) = \sum_{k=0}^{e-1} m_{i,j,k}\zeta^k$ where $m_{i,j,k}$ are integers satisfying $0 \leqslant m_{i,j,k} \leqslant |\chi_i(g_j)| \leqslant \chi_i(1) < p$. The following lemma will help us to determine such coefficients.

Lemma 2.2.5. *If* $\chi_i(g_j) = \sum_{k=0}^{e-1} m_{i,j,k}\zeta^k$, *then*

$$m_{i,j,k} = e^{-1} \sum_{l=0}^{e-1} \chi_i(g_j^l)\zeta^{-kl}. \tag{2.8}$$

Proof. For any $l \in \mathbb{Z}$ we have that $\chi_i(g_j^l) = \sum_{t=0}^{e-1} m_{i,j,t}\zeta^{tl}$. Then,

$$e^{-1} \sum_{l=0}^{e-1} \chi_i(g_j^l)\zeta^{-kl} = e^{-1} \sum_{t=0}^{e-1} m_{i,j,t} \sum_{l=0}^{e-1} \zeta^{(t-k)l}.$$

As ζ is a primitive e-th root of unity, the sum $\sum_{l=0}^{e-1} \zeta^{(t-k)l}$ equals e if $t = k$ and 0 otherwise. We get the desired equality. $\square$

Applying Θ to equation (2.8), we get the value of $m_{i,j,k}$ mod p:

$$m_{i,j,k} \equiv_p \Theta(e^{-1}) \sum_{l=0}^{e-1} \Theta(\chi_i(g_j^l))\omega^{-kl}.$$

Since the matrix $\Theta(X)$ is known, it is clear that the values of $\Theta(\chi_i(g_j^l))$ can be computed. The value $\Theta(\chi_i(g_j^l))$ equals to the $(f(g_j^l), i)$-th entry of the matrix $\Theta(X)$, where f denotes the class map with respect to the conjugacy classes we have fixed at the beginning. Finally, as $0 \leqslant m_{i,j,k} \leqslant \chi_i(1) < p$, the values of the integers $m_{i,j,k}$ are fully determined, and hence the character table of G is known.

The following is a summary in pseudocode of the complete algorithm:

Algorithm 2: Burnside-Dixon-Schneider

```
Data: $G$ finite group with $1 = g_1, \dots, g_r$ representatives of its
      conjugacy classes. $f$ denotes the class map of $G$ with respect
      to these representatives.
Result: The character table of $G$ as a matrix.
1  $e \coloneqq \operatorname{lcm}(g_1, \dots, g_r)$;
2  Choose a prime number $p$ with $e|p-1$ and $p > 2\sqrt{|G|}$;
3  Find $v_1, \dots, v_r$ linearly independent common eigenvectors of the
     matrices $\Theta(M_j)$, $1 \leqslant j \leqslant r$, using Algorithm 1;
4  for $1 \leqslant i \leqslant r$ do
       // Determine the values of the degrees $\chi_i(1)$
5      Compute the unique integer satisfying $1 \leqslant d_i < p/2$ and
         $\sum_{j=1}^{r} |C_j| v_i[j] v_i[j'] \equiv_p |G|/d_i^2$;
6  for $1 \leqslant i \leqslant r$ do
       // Compute the matrix $\Theta(X)$ row by row
7      $\Theta(X)_{i,:} \coloneqq d_i v_i$;
8  for $1 \leqslant i, j, k \leqslant r$ do
9      $m_{i,j,k} \coloneqq \Theta(e^{-1}) \sum_{l=0}^{e-1} \Theta(X)_{f(g_j^l),i}\ \omega^{-kl}$;
10 for $1 \leqslant i, j \leqslant r$ do
11     $\chi_{i,j} \coloneqq \sum_{k=0}^{e-1} m_{i,j,k} \zeta^k$;
12 return $X = (\chi_{i,j})_{1 \leqslant i,j \leqslant r}$;
```

Note: The author's implementation of this algorithm in the GAP system [GAP22] can be found in the file `CharTab.g` from the GitHub repository [Jua23] and in Appendix B.

The algorithm described in this section can be improved in many ways. For example, one could first compute the linear characters and use this information to avoid computing class matrices. The interested reader should check [Hul93] for more details.

Chapter 3

Ore's conjecture

Ore's conjecture states that every element of every finite non-abelian simple group is a commutator. In 2009 the proof was completed.

3.1 Generalizations

Obvious generalizations of Ore's conjecture fail to hold. For example, in exercise A.2.1 we prove that for every prime number p there is a group G of order p^{12} such that $G' \neq K(G)$, where $K(G)$ denotes the set of commutators of G. Also, in exercise A.2.2 we check in the GAP system [GAP22] that the smallest group in which the property $K(G) \neq G'$ holds has order 96.

Now we note an open problem related to Ore's conjecture. There is a conjecture attributed to J. G. Thompson which asserts that if G is a finite non-abelian simple group, then there is a conjugacy class $C \subseteq G$ with $G = C^2$. It is straightforward to check that Thompson's conjecture implies Ore's conjecture, as we do now. Let $C = \mathrm{Cl}_G(g)$. Since $1 \in G = C^2$, we have that $g^{-1} \in C$, and thus $G = \mathrm{Cl}_G(g^{-1})\,\mathrm{Cl}_G(g)$. Hence, every element of G is of the form $w = (g^{-1})^x g^y$, and consequently $w = (h^{-1})^z h$ is a commutator, where $z = y^{-1}x$ and $h = g^y$.

We remark that a weaker form of Thompson's conjecture is true. If G is a finite non-abelian simple group, in [GT15] was proved that there exists a conjugacy class $C \subseteq G$ with $G = C^3$.

3.2 Alternating groups

The alternating groups are the "simplest" family of the simple non-abelian groups. In the present section we prove Ore's conjecture for this particular groups.

We use the usual convention for the product of two permutations, i.e. the product of two $\sigma\tau$ permutations will be the composition where we apply σ first and then τ. Also, we write $(x)\sigma$ to denote the image of an element

x under the permutation σ. We say that $(a_1 \dots a_i) \dots (a_j \dots a_n)$ *involves elements from* $(b_1 \dots b_l) \dots (b_r \dots b_m)$ when $\{a_i\}_{i=1}^n \subseteq \{b_i\}_{i=1}^m$.

We start, as always, with some technical lemmas.

Lemma 3.2.1. *Let G be a group. If $a_1, b_1, a_2, b_2 \in G$ and a_1 and b_1 commute with both a_2 and b_2, then $[a_1, b_1][a_2, b_2] = [a_1 a_2, b_1 b_2]$.*

Proof. First, recall that if $x, y, z \in G$, then $[xz, y] = [x, y]^z[z, y]$ and $[x, zy] = [x, y][x, z]^y$. Therefore,

$$\begin{aligned}[a_1a_2, b_1b_2] &= [a_1, b_1b_2]^{a_2}[a_2, b_1b_2]\\ &= ([a_1, b_2][a_1, b_1]^{b_2})^{a_2}[a_2, b_2][a_2, b_1]^{b_2} = [a_1, b_1][a_2, b_2].\end{aligned}$$

□

Lemma 3.2.2. *Let $c_1, c_2 \in A_n$ be two cycles of the same length, i.e. $c_1 = (a_1 \dots a_m)$ and $c_2 = (b_1 \dots b_m)$ for some $3 \leqslant m \leqslant n$. If c_2 fixes a_i and a_j with $a_i \neq a_j$, then there exists $\varphi \in A_n$ such that $c_1^{-1} = c_2^{\varphi}$ and that the a_i's and b_i's appear in its decomposition.*

Proof. As c_1^{-1} and c_2 are cycles of the same length, there exists $\varphi \in S_n$ such that $c_1^{-1} = c_2^{\varphi}$. If $\varphi \in A_n$, the first part of the claim is proven. Otherwise,

$$c_2^{\varphi} = ((b_1)\varphi \dots (b_m)\varphi) = c_1^{-1} = (a_m \dots a_1). \tag{3.1}$$

Therefore, as $a_i, a_j \notin \{b_i\}_{i=1}^m$, we conclude that $c_1^{-1} = c_2^{(a_i a_j)\varphi}$ and $(a_i a_j)\varphi \in A_n$.

From (3.1) we deduce that we can suppose that any element different to the a_i-s and b_i-s can be fixed by φ, thus, they only appear a_i's and b_i's in φ's decomposition. □

Now we are ready to prove our desired theorem. There may be shorter proves of it, but the one we present is interesting since it is constructive.

Theorem 3.2.3. *When $n \geqslant 5$, every element of A_n is a commutator.*

Proof. First, we show that some particular permutations of A_n are commutators of two permutations that involve elements from the given permutation. Afterwards, we will see that this is sufficient to prove the theorem. We start writing cycles of odd order (greater than 3) as commutators.

Case 1.1: If the order of the cycle is $2m+1$, for some $m \geqslant 2$ and m even, then

$$\begin{aligned} c &= (a_1 a_2 a_3 \dots a_{m+1} a_{m+2} a_{m+3} \dots a_{2m+1}) \qquad (3.2)\\ &= (a_1 a_2 a_3 \dots a_{m+1})(a_1 a_{m+2} a_{m+3} a_{m+4} \dots a_{2m+1}). \qquad (3.3)\end{aligned}$$

If we name c_1 the cycle on the left of the equation (3.3) and c_2 the one on the right, it is clear that they are of the same length, $m+1$; and that $c_1, c_2 \in A_n$.

Furthermore, c_2 fixes a_2 and a_3, consequently, by Lemma 3.2.2 there exists $\varphi \in A_n$ such that $c_1^{-1} = c_2^{\varphi}$, hence, $c = c_1c_2 = [\varphi, c_2]$. Remember that from Lemma 3.2.2 we also know that φ will involve elements from c.

Case 1.2: When the order of the cycle is $2m+1$, for some $m \geqslant 3$ and m odd, then

$$\begin{aligned} c &= (a_1a_2a_3 \dots a_{m+1}a_{m+2}a_{m+3} \dots a_{2m+1}) \\ &= (a_1a_{m+2}a_2 \dots a_{m+1})(a_1a_{m+2}a_2a_{m+3} \dots a_{2m+1}). \end{aligned}$$

Defining c_1 and c_2 as in the previous case, c_2 fixes a_3 and a_4; c_1 and c_2 are of length $m+2$; and $c_1, c_2 \in A_n$. Thus, by Lemma 3.2.2 there exists $\varphi \in A_n$ such that $c = c_1c_2 = [\varphi, c_2]$. Remember that from Lemma 3.2.2 we also know that φ will involve elements from c.

Now, we consider the case where we have a pair of disjoint cycles of even order. Suppose that we have the product of a cycle of order $2k$ with another of order $2m-2k$ for some $k, m \in \mathbb{N}$ such that $0 < 2k \leqslant m$.

Case 2.1: When $m = 2$, the unique possibility for k is being 1, and the following proves the claim:

$$(a_1a_2)(a_3a_4) = (a_1a_2a_3)(a_1a_4a_3) = [(a_1a_3)(a_2a_4), (a_1a_4a_3)]. \tag{3.4}$$

Case 2.2: If m is even and $m \geqslant 4$:

$$\begin{aligned} c &= (a_1a_2 \dots a_{2k})(a_{2k+1}a_{2k+2} \dots a_{m+1}a_{m+2}a_{m+3} \dots a_{2m}) \\ &= (a_1a_2 \dots a_{m+1})(a_1a_{m+2}a_{m+3} \dots a_{2m}a_{2k+1}) \end{aligned}$$

Defining c_1 and c_2 as always, c_2 fixes a_2 and a_m. Evidently, both are of length $m+1$, so $c_1, c_2 \in A_n$. Therefore, by Lemma 3.2.2 there exists $\varphi \in A_n$ such that $c = c_1c_2 = [\varphi, c_2]$.

Case 2.3: If $m = 3$ and $k = 1$ we have that:

$$\begin{aligned} (a_1a_2a_3a_4)(a_5a_6) &= (a_2a_3a_5a_4a_6)(a_1a_2a_5a_4a_6) \\ &= [(a_1a_6a_2a_4a_3), (a_1a_2a_5a_4a_6)]. \end{aligned}$$

Case 2.4: When m is odd and $m \geqslant 3$ but $(m, k) \neq (3, 1)$,

$$\begin{aligned} c &= (a_1a_2a_3 \dots a_{2k})(a_{2k+1}a_{2k+2} \dots a_{m+1}a_{m+2}a_{m+3} \dots a_{2m}) \\ &= (a_1a_{m+2}a_2 \dots a_{m+1})(a_1a_{m+2}a_2a_{m+3} \dots a_{2m}a_{2k+2}) \end{aligned}$$

Defining c_1 and c_2 as always, we see that c_2 fixes a_4 and a_{m+1}. It is easy to see that both are of length $m+2$, so $c_1, c_2 \in A_n$. Therefore, by Lemma 3.2.2 there exists $\varphi \in A_n$ such that $c = c_1c_2 = [\varphi, c_2]$.

In the following lines we consider the cases when we deal with 3-cycles.

Case 3.1: Let $\tau = (a_1a_2a_3) \dots (a_ia_{i+1}a_{i+2})$ be the product of an even number of 3-cycles. Our aim is to find $\pi_1, \pi_2, \varphi \in A_n$ such that $\tau = \pi_1\pi_2$

and $\pi_1^{-1} = \pi_2^{\varphi}$. As for any $j \leqslant i$, $(a_j a_{j+1} a_{j+2}) = (a_j a_{j+2} a_{j+1})^2$ and the 3-cycles are disjoint, by defining $\pi_1 = \prod_{j=1}^{i}(a_j a_{j+2} a_{j+1}) = \pi_2$, we have that $\tau = \pi_1\pi_2$. So, there exists $\varphi \in S_n$ such that $\pi_1^{-1} = \pi_2^{\varphi}$, and then

$$\pi_1^{-1} = (a_2 a_3 a_1)\dots(a_{i+1} a_{i+2} a_i) \tag{3.5}$$
$$\pi_2^{\varphi} = ((a_1)\varphi(a_3)\varphi(a_2)\varphi)\dots((a_i)\varphi(a_{i+2})\varphi(a_{i+1})\varphi). \tag{3.6}$$

As (3.5) equals to (3.6) we conclude that $\varphi = (a_1 a_2)\dots(a_i a_{i+1})$ works and that $\varphi \in A_n$ (as the number of 3-cycles is even); and therefore, $\tau = [\varphi, \pi_2]$. Observe that both φ and π_2 only involve elements from τ.

Case 3.2: Now, we study the case when we have a product of an odd (greater than 1) number of 3-cycles. Due to the previous case we know that if $\tau = (a_1 a_2 a_3)\dots(a_i a_{i+1} a_{i+2})$, then $\tau = [\varphi, \pi_2]$, where φ and π_2 are the same permutations from Case 3.1. Unfortunately, as the number of 3-cycles is odd, $\varphi \notin A_n$. Since the number of 3-cycles is greater than 1 and recalling both (3.5) and (3.6), we have that $\tau = [(a_1 a_i)(a_3 a_{i+2})(a_2 a_{i+1})\varphi, \pi_2]$ which gives the desired result.

Case 3.3: In this case we will write $\tau = (a_1 a_2 a_3)(a_4 a_5)(a_6 a_7)$ as a commutator. Recalling the decomposition from (3.4) and as $(a_1 a_2 a_3) = (a_1 a_3 a_2)^2$, by taking $\pi_1 = (a_1 a_3 a_2)(a_4 a_5 a_6)$ and $\pi_2 = (a_1 a_3 a_2)(a_4 a_7 a_6)$, $\tau = \pi_1\pi_2$. It is clear that there exists $\varphi \in S_n$ such that $\pi_1^{-1} = \pi_2^{\varphi}$. Let's suppose that $\varphi \notin A_n$.

$$\pi_1^{-1} = (a_2 a_3 a_1)(a_6 a_5 a_4)$$
$$\pi_2^{\varphi} = ((a_1)\varphi(a_3)\varphi(a_2)\varphi)((a_4)\varphi(a_7)\varphi(a_6)\varphi)$$

whence we see that $(a_1 a_4)(a_3 a_7)(a_1 a_6)\varphi \in A_n$ and $\pi_1^{-1} = \pi_2^{(a_1 a_4)(a_3 a_7)(a_1 a_6)\varphi}$. So, in any case, there is some $\psi \in A_n$ such that $\tau = [\psi, \pi_2]$. Also, it is clear that ψ will only involve elements from τ.

Case 3.4: We will express $\tau = (a_1 a_2 a_3)(a_4 a_5)(a_6 a_7 a_8 a_9)$ as a commutator. Using the decomposition $(a_4 a_5)(a_6 a_7 a_8 a_9) = (a_5 a_6 a_7 a_9 a_8)(a_4 a_5 a_9 a_8 a_6)$ and proceeding as in the previous case,

$$\tau = (a_1 a_3 a_2)(a_1 a_3 a_2)(a_5 a_6 a_7 a_9 a_8)(a_4 a_5 a_9 a_8 a_6).$$

Defining $\pi_1 = (a_1 a_3 a_2)(a_5 a_6 a_7 a_9 a_8)$ and $\pi_2 = (a_1 a_3 a_2)(a_4 a_5 a_9 a_8 a_6)$, we see that $\tau = \pi_1\pi_2$. Moreover,

$$\pi_1^{-1} = (a_2 a_3 a_1)(a_8 a_9 a_7 a_6 a_5)$$
$$\pi_2^{\varphi} = ((a_1)\varphi(a_3)\varphi(a_2)\varphi)((a_4)\varphi(a_5)\varphi(a_9)\varphi(a_8)\varphi(a_6)\varphi)$$

from where we deduce that by taking $\varphi = (a_1 a_2)(a_4 a_8 a_6 a_5 a_9 a_7) \in A_n$, we get that $\tau = [\varphi, \pi_2]$.

At this point we are ready to show that any permutation $\sigma \in A_n$ is a commutator. If the amount of 3-cycles of σ in its decomposition is different from 1, then (by cases 1.1, 1.2, 2.1, 2.2, 2.3, 2.4, 3.1 and 3.2) σ is the product of some commutators, $\sigma = \prod_i [\varphi_i, \psi_i]$. Since every φ_i and ψ_i involve elements from $[\varphi_i, \psi_i]$, by Lemma 3.2.2 we conclude that σ is a commutator.

Suppose that σ contains only one 3-cycle in its decomposition and that σ it is not a 3-cycle. If in σ's decomposition there are some factors of the type from cases 3.3 or 3.4, by arguing as in the previous paragraph we get that σ is a commutator. Otherwise, in σ's decomposition there is at least one factor of the type from cases 1.1, 1.2, 2.2 or 2.4. Write $\sigma = (a_1a_2a_3)[\varphi, \psi]$, for some $\varphi, \psi \in A_n$. It can be easily shown (using the same idea from the proof of Lemma 3.2.2) that $\sigma = [(a_1a_2)\varphi, (a_1a_3a_2)\psi]$. Since in cases 1.1, 1.2, 2.2 and 2.4 we have applied Lemma 3.2.2, rearranging its proof we can suppose that $\varphi \in S_n - A_n$, and hence we get the desired result.

Finally, recalling that $n \geqslant 5$ by hypothesis, we have that any 3-cycle is a commutator

$$(a_1a_2a_3) = (a_2a_5a_1)(a_2a_4a_3)(a_1a_5a_2)(a_3a_4a_2) = [(a_1a_5a_2), (a_3a_4a_2)].$$

□

Following the idea from the proof the author has implemented[1] an algorithm in the GAP system [GAP22] which receives as input a permutation σ of A_n and outputs $\varphi, \psi \in A_n$ such that $\sigma = [\varphi, \psi]$. Note that the choice of φ and ψ is not unique. In fact, in a group G, for any $s, t \in G$ we have that $[s,t] = [rs,t]$ if and only if $r \in C_G(t)$. At the time of writing this dissertation, the author has not found any function or package for the GAP system [GAP22] that does the same as the algorithm described above.

3.3 Testing Ore's conjecture

The main goal of the dissertation is to reproduce the computations behind the proof of Main Lemma and understand some of the algorithms behind them. The naive test used in Exercise A.2.2 is not effective when dealing with big groups (order $n \geqslant 10^6$), as the memory use is huge. However, the tests we present in this dissertation will perform well when dealing with big groups, as we will see at the end of the dissertation.

Combining Algorithm 2 from the previous chapter (the one that computes character tables) with Theorem 2.1.4 (Ore's criterion) we get a test to check whether every element from a group is a commutator. We also use the elementary fact that a conjugate of a commutator is again a commutator. In fact, this test was used in [LOST]. The following is a summary of the

[1]It can be found in the file `AsCommutator.g` from the GitHub repository [Jua23] and in Appendix B.

test written in pseudocode:

Test A: Ore (deterministic)

```
  Data: Finite group $G$.
1 Compute the character table $X = (\chi_i(g_j))$ of $G$ using Algorithm 2;
2 for $1 \leqslant j \leqslant r$ do
3     if $\sum_{i=1}^{r} \chi_i(g_j)/\chi_i(1) = 0$ then
4         return False;
5 return True;
```

Note: The author's implementation of this test in the GAP system [GAP22] can be found in the file `OreTest1.g` from the GitHub repository [Jua23] and in Appendix B.

The test returns `True` when every element of a group G is a commutator and `False` otherwise. Moreover, the test always terminates and the runtime does not change given the same group as input, as long as the same happens to the algorithm from the first step.

3.4 Testing Ore's conjecture probabilistically

In contrast with the deterministic test given in the previous chapter, in the following lines we present a probabilistic test to check Ore's conjecture for a concrete finite group. The correctness of the test relies on two elementary facts: every conjugate of a commutator is again a commutator, and on the class equation

$$|G| = \sum_i |G : C_G(g_i)|,$$

where we choose a single representative element g_i from each conjugacy class.

We may assume that we know a way of choosing random elements from a group (see [Cel+95]) and how to compute the centralizer of an element (see [HEO05, §4.6.5]). In order to test if every element in a group is a commutator we can proceed in the following way. We choose random elements that are commutators, test them for conjugacy with the known class representatives and continue until we get a complete set of representatives of the conjugacy classes. Completeness is checked by computing the index of the centralizers of the representative and applying the class equation.

We may assume that the group we are given is a permutation group, since storing groups in a computer as permutation groups is common. Thanks to this imposition we can save time in the following way: before checking conjugacy directly in the given group we will compare the cycles types of the elements, that is, check conjugacy in the symmetric group. Therefore, if two elements result to not be conjugate in the symmetric group, in particular they will not be conjugate in the given group. We apply this trick since in practice it is by far faster to compare the cycle type of two elements than checking whether two elements are conjugate in a group. We summarize the test in pseudocode:

Test B: Ore (probabilistic)

```
   Data: Finite permutation group $G \leqslant S_n$.
1  $\mathcal{R} \coloneqq \{1\}$;
   // In this set we store the different representatives of the
      conjugacy classes of $G$
2  while $|G| > \sum_{x\in\mathcal{R}} |G : C_G(x)|$ do
3      $g \coloneqq$ [Random($G$), Random($G$)];
4      for $x \in \mathcal{R}$ do
5          if IsConjugate($S_n, x, g$) = True then
               // In fact we should check that whether $x$ and $g$ have
                  the same cycle type or not
6              if IsConjugate($G, x, g$) = True then
7                  break;
8      else
           // If $g$ is not conjugate to any $x \in \mathcal{R}$
9          $\mathcal{R} \coloneqq \mathcal{R} \cup \{g\}$;
10 return True;
```

Note 1: The author's implementation of this test in the GAP system [GAP22] can be found in the file OreTest2.g from the GitHub repository [Jua23] and in Appendix B.
Note 2: Random(G) represents a uniformly and randomly chosen element from G and IsConjugate(G, x, y) is a function that checks whether x and y are conjugate in G or not.

Note that this test never returns False and that it terminates if and only if every element of the group is a commutator. Since we choose random elements from the group, for the same input the runtime of the test may be different. Algorithms of this type are called *Las Vegas algorithms*. They always output correct results. However, they may make random choices during their execution, causing their runtime to fluctuate between executions, even when provided with identical inputs. Other common class of probabilistic algorithms are *Monte Carlo algorithms*. In contrast with the Las Vegas algorithms, they always terminate in a finite number of time, but the output of them may be inaccurate, usually with a low probability.

Finally, observe that by changing Line 3 by $g \coloneqq$ Random(G) the test is converted to an algorithm that computes the representatives of the classes of a finite group. A test of this type results to be very efficient in simple groups, since the number of conjugacy classes in simple groups tends to be small. For a more detailed discussion on algorithms for computing conjugacy classes, we refer the interested readers to [CS97].

3.5 Experimental results and final discussion

In this section we will present the experimental results on the performance of different tests that determine Ore's conjecture for a concrete group. We have chosen part of the groups that where checked in [LOST, Lemma 3.1], Main Lemma from the introduction.

All the results are displayed in Table 3.1, where we provide information about the order of the different groups and the number of conjugacy classes of each one. The most interesting part of the table (for this dissertation) are the running times of the different tests. We refer to the different tests as follows:

- Test A.1: the deterministic test, computing the character tables using GAP's default implementation[2].
- Test A.2: the deterministic test, computing the character tables using the author's implementation of Algorithm 2.
- Test B: the probabilistic test discussed in Section 3.4.

Also, in Table 3.2 (p. 39), we present the results when applying the test to some simple groups.

Running times marked by — were not finished after 2100 seconds. The groups that we have omitted from Main Lemma surpassed this time in all the tests. In the column that corresponds to the CPU time of the Test B we write two values: the mean and the standard deviation of the CPU time of five executions. When the mean is smaller than one second, we simply write < 1. For the other tests, only a computation was performed. Furthermore, in the last column we write Yes if the group fulfills Ore's conjecture.

Final discussion

In most of the cases the fastest test has been Test A.1, and for all the groups it has performed better than Test A.2. In fact, the implementation of the algorithm that computes the character table in Test A.1 contains several optimizations that the author of this dissertation has not included in his implementation. We also note that GAP's implementation of such algorithm has a length 2633 lines of code, whereas the implementation written for this dissertation has 313 lines, which should make the latter easier to understand.

As one would expect, the smaller the number of conjugacy classes is, the better performance by the random test. For example, $\Omega_8^-(2)$ has 53 classes. For this group Test A.1 surpassed the limit time of 2100 seconds and Test B performed in a modest amount of time. In contrast, when applying Test B to $\mathrm{Sp}_6(3)$, which has 141 classes, the runtime surpassed the established limit in the five executions.

Furthermore, another factor plays an important role in the performance of Test B: the sizes of the conjugacy classes. If a group results to have a

[2]The implementation can be found on https://github.com/gap-system/gap/blob/master/lib/ctblgrp.gi

Table 3.1: Experimental results
Some groups from Main Lemma

Group	Order	Classes	CPU time (sec)			Ore
			Test A.1	Test A.2	Test B	
$\mathrm{Sp}_6(2)$	$2^9 \cdot 3^4 \cdot 5 \cdot 7$	30	1.9	34.3	$\mu = 38$, $\sigma = 29$	Yes
$\mathrm{Sp}_8(2)$	$2^{16} \cdot 3^5 \cdot 5^2 \cdot 7 \cdot 17$	81	28.2	—	—	Yes
$\mathrm{Sp}_4(3)$	$2^7 \cdot 3^4 \cdot 5$	34	< 1	2.5	$\mu = 10.24$, $\sigma = 5.2$	Yes
$\mathrm{Sp}_6(3)$	$2^{10} \cdot 3^9 \cdot 5 \cdot 7 \cdot 13$	141	85	—	—	Yes
$\mathrm{SU}_3(3)$	$2^5 \cdot 3^3 \cdot 7$	14	< 1	2.8	< 1	Yes
$\mathrm{SU}_3(4)$	$2^6 \cdot 3 \cdot 5^2 \cdot 13$	22	1.4	29.2	$\mu = 2.28$, $\sigma = 0.68$	Yes
$\mathrm{SU}_3(5)$	$2^4 \cdot 3^3 \cdot 5^3 \cdot 7$	40	5.7	1096	—	Yes
$\mathrm{SU}_3(7)$	$2^7 \cdot 3 \cdot 7^3 \cdot 43$	58	2040	—	$\mu = 169.6$, $\sigma = 132.98$	Yes
$\mathrm{SU}_4(2)$	$2^6 \cdot 3^4 \cdot 5$	20	< 1	< 1	$\mu = 3.6$, $\sigma = 1.26$	Yes
$\mathrm{SU}_4(3)$	$2^9 \cdot 3^6 \cdot 5 \cdot 7$	71	18.7	—	—	Yes
$\mathrm{SU}_5(2)$	$2^{10} \cdot 3^5 \cdot 5 \cdot 11$	47	6.4	—	$\mu = 1081.2$, $\sigma = 664.4$	Yes
$\mathrm{SU}_6(2)$	$2^{15} \cdot 3^7 \cdot 5 \cdot 7 \cdot 11$	132	1070	—	—	Yes
$\Omega_8^+(2)$	$2^{12} \cdot 3^5 \cdot 5^2 \cdot 7$	53	13.6	—	$\mu = 138$, $\sigma = 42$	Yes
$\Omega_8^-(2)$	$2^{12} \cdot 3^4 \cdot 5 \cdot 7 \cdot 17$	39	—	—	$\mu = 210$, $\sigma = 195$	Yes
$\Omega_7(3)$	$2^9 \cdot 3^9 \cdot 5 \cdot 7 \cdot 13$	58	15.8	—	—	Yes

Note 1: All times have been obtained on an MacBook Pro running macOS 12.2.1 with a 2,5 GHz Quad-Core Intel Core i7 processor and 16 GB of RAM.
Note 2: The orders and the number of classes have been computed using the GAP system [GAP22].

small conjugacy class, since we are assuming that we pick random elements from the group uniformly, it would be very unlikely to find an element in such a class. This can slow down greatly the performance of the random test. For instance, the runtime exceeded the limit when applying Test B to the groups $\mathrm{SU}_3(5)$, $\mathrm{SU}_4(3)$ and $\mathrm{SU}_6(2)$. The reason of this is that in those three groups there are non-trivial classes of size 1. For example, the probability of finding an element randomly in one of those classes in the group $\mathrm{SU}_6(2)$ is similar to the probability of winning the biggest prize in the Spanish lottery in two years in a row.

If we just focus in the CPU time of the experiments performed with the random method, we can observe that the times have a big standard deviation. This is a consequence of the random nature of the test.

As a conclusion, we show a heuristic to choose the optimal test for checking Ore's conjecture for a concrete group. The first step is to compute the conjugacy classes of the given group. If there is a class of size 1 (or small enough at the user's discretion) Test B should be discarded and Test A.1 should be applied. If all classes turn out to be of similar size, and the number is small, applying Test B seems sensible. This normally occurs in simple groups, in particular in sporadic groups. However, all the character tables of the sporadic groups are known and have been stored. Therefore, when dealing with these groups the best option is to apply Test A (the deterministic test) to the precomputed character tables, which are available, for instance, in the GAP package CTblLib [Bre22].

Table 3.2: Experimental results

Some simple groups

Group	Order	Classes	CPU time (sec) Test A.1	Test A.2	Test B	Ore
$\mathrm{PSL}_3(2)$	$2^3 \cdot 3 \cdot 7$	6	< 1	< 1	< 1	Yes
$\mathrm{PSL}_3(3)$	$2^4 \cdot 3^3 \cdot 13$	12	< 1	2.12	< 1	Yes
$\mathrm{PSL}_5(2)$	$2^{10} \cdot 3^2 \cdot 5 \cdot 7 \cdot 31$	27	32	—	$\mu = 6.7$, $\sigma = 5.4$	Yes
$\mathrm{PSp}_4(3)$	$2^6 \cdot 3^4 \cdot 5$	20	< 1	< 1	$\mu = 3.06$, $\sigma = 0.91$	Yes
$\mathrm{PSp}_6(3)$	$2^9 \cdot 3^9 \cdot 5 \cdot 7 \cdot 13$	74	21.5	—	—	Yes
$\mathrm{PSU}_4(3)$	$2^7 \cdot 3^6 \cdot 5 \cdot 7$	20	5.7	—	$\mu = 9.2$, $\sigma = 6.87$	Yes
M_{11}	$2^4 \cdot 3^2 \cdot 5 \cdot 11$	10	< 1	2.2	< 1	Yes
M_{12}	$2^6 \cdot 3^3 \cdot 5 \cdot 11$	15	< 1	5.5	< 1	Yes
M_{22}	$2^7 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11$	12	< 1	190	< 1	Yes
Suz	$2^{13} \cdot 3^7 \cdot 5^2 \cdot 7 \cdot 11 \cdot 13$	43	< 1	—	—	Yes
$\mathbb{M}$	$2^{46} \cdot 3^{20} \cdot 5^9 \cdot 7^6 \cdot 11^2 \cdot 13^3 \cdot 17 \cdot 19 \cdot 23 \cdot 29 \cdot 31 \cdot 41 \cdot 47 \cdot 59 \cdot 71$	194	< 1	—	—	Yes
J_4	$2^{21} \cdot 3^3 \cdot 5 \cdot 7 \cdot 11^3 \cdot 23 \cdot 29 \cdot 31 \cdot 37 \cdot 43$	62	< 1	—	—	Yes

Note 1: All times have been obtained on an MacBook Pro running macOS 12.2.1 with a 2,5 GHz Quad-Core Intel Core i7 processor and 16 GB of RAM.

Note 2: The orders and the number of classes have been computed using the GAP system [GAP22].

Note 3: For the sporadic groups Test A.1 means applying Test A to the precomputed character tables.

Appendix A

Exercises

A.1 Exercises from Chapter 1

Exercise A.1.1. Compute the orders of the linear groups.

Solution. An invertible matrix takes a basis to another basis and is determined by the image of an ordered basis. Only a condition is imposed to this image: the image of the i-th vector must be linearly independent to all the previous ones. Since the previous $i-1$ vectors generate a subgroup of dimension $i-1$, which has q^{i-1} vectors in it, we get that

$$\begin{aligned} |\operatorname{GL}_n(q)| &= (q^n-1)(q^n-q)(q^n-q^2)\dots(q^n-q^{n-1}) \\ &= q^{n(n-1)/2}(q-1)(q^2-1)\dots(q^n-1). \end{aligned}$$

From the definition of $\operatorname{SL}_n(q)$ it is clear that $|\operatorname{SL}_n(q)| = \dfrac{|\operatorname{GL}_n(q)|}{q-1}$. Finally, to get $|\operatorname{PSL}_n(q)|$ we need to count the number of scalar matrices λI_n with determinant 1. Since the determinant of these scalar matrices is $\lambda^n = 1$ and the number of solutions to $x^n = 1$ in $\mathbb{F}_q$ is $(n, q-1)$,

$$|\operatorname{PSL}_n(q)| = \frac{1}{(n,q-1)} q^{n(n-1)/2}(q-1)(q^2-1)\dots(q^n-1).$$

□

Exercise A.1.2. Compute the orders of the symplectic groups.

Solution. Let V be a $\mathbb{F}_q$-vector space of dimension $2m$. By definition, $\operatorname{Sp}(V)$ acts faithfully and transitively on the set of ordered symplectic bases of V. Hence, $|\operatorname{Sp}_{2m}(q)|$ equals the number of such bases.

We start with e_1 which can be any non-zero vector, so there are $q^{2m}-1$ ways of choosing it. Since $e_1^{\perp}$ has dimension $2m-1$, it has q^{2m-1} vectors. Hence, the number of vectors v with $f(u,v) \neq 0$ is $q^{2m} - q^{2m-1} = (q-1)q^{2m-1}$. These come in sets of $q-1$ scalar multiples, each one giving a different value $f(u,v)$. Thus, the number of ways of choosing f_1 is q^{2m-1}, and by induction on m, we get that

$$|\operatorname{Sp}_{2m}(q)| = \prod_{i=1}^{m}(q^{2i}-1)q^{2i-1}.$$

Clearly, $|\operatorname{PSp}_{2m}(q)| = \dfrac{|\operatorname{Sp}_{2m}(q)|}{(2,q-1)}$. □

Exercise A.1.3. Prove that the transvections in $\operatorname{Sp}_4(3)$ and $\operatorname{Sp}_6(2)$ are commutators.

Solution. Let $V \cong \mathbb{F}_q^{2m}$ be a vector space. Since the conjugate of a commutator is a commutator again, $\operatorname{Sp}(V)$ acts on $V - \{0\}$ transitively and by the property iv) from Proposition 1.4.2, it suffices to verify that the transvections $T_v(\lambda)$, for a fixed vector v, are a commutator for every scalar λ. Let $\mathcal{B} = \{e_1, \dots, f_m\}$ be a symplectic basis of V. Then, it is clear that $M_{\mathcal{B}}(T_{f_1}(\lambda)) = \left(\begin{smallmatrix} I_m & C \\ 0 & I_m \end{smallmatrix}\right)$ where $C \in M_m(\mathbb{F}_q)$ and all the entries of C are zero except the entry (1,1), whose value is λ.

From (1.5) we deduce that if $A \in \operatorname{GL}_m(q)$ and B is a $m \times m$ symmetric matrix, then $M_A = \left(\begin{smallmatrix} A^{-1} & 0 \\ 0 & A^T \end{smallmatrix}\right) \in \operatorname{Sp}_{2m}(q)$ and $M_B = \left(\begin{smallmatrix} I_m & B \\ 0 & I_m \end{smallmatrix}\right) \in \operatorname{Sp}_{2m}(q)$. An easy computation shows that $[M_A, M_B] = \left(\begin{smallmatrix} I_m & B-ABA^T \\ 0 & I_m \end{smallmatrix}\right)$.

The following choices establish the result. When $m = 2$ and $q = 3$, and if $\lambda \in \mathbb{F}_3^\times$,

$$A = \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

When $m = 3$ and $q = 2$,

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

□

Exercise A.1.4. Check in GAP that $\operatorname{PSU}_3(2)$ is a soluble group.

Solution. The following code solves the exercise:

```
gap> IsSolvableGroup(ProjectiveSpecialUnitaryGroup(3,2));
true
```

□

A.2 Exercises from Chapter 3

Exercise A.2.1. [Rot95, p. 34]

(i) Let $k[x,y]$ denote the ring of all polynomials in two variables over a field k, and let $k[x]$ and $k[y]$ denote the subrings of all polynomials in x and in y, respectively. Define G to be the set of all matrices of the form

$$(f,g,h) \coloneqq \begin{pmatrix} 1 & f & h \\ 0 & 1 & g \\ 0 & 0 & 1 \end{pmatrix}$$

where $f \in k[x]$, $g \in k[y]$ and $h \in k[x,y]$[1]. Prove that G is a multiplicative group and that G' consists of all matrices for which $f = 0 = g$.

(ii) If $(0,0,h)$ is a commutator, then there are polynomials $f_1, f_2 \in k[x]$ and $g_1, g_2 \in k[y]$ with $h = f_1g_2 - f_2g_1$.

(iii) Show that if $h(x,y) = x^2 + xy + y^2$, then $(0,0,h) \in G'$ is not a commutator.

(iv) Deduce that for every prime number p there is a group G of order p^{12} such that $G' \neq K(G)$.

Solution. (i) Let $(f_1,g_1,h_1),(f_2,g_2,h_2) \in G$, then as a consequence of the multiplication of matrices

$$(f_1,g_1,h_1)(f_2,g_2,h_2) = (f_1+f_2, g_1+g_2, h_1+h_2+f_1g_2) \qquad \text{(A.1)}$$
$$(f_1,g_1,h_1)^{-1} = (-f_1, -g_1, f_1g_1 - h_1) \qquad \text{(A.2)}$$

whence we get that $G \leqslant \mathrm{GL}_3(k[x,y])$, and thus G is a group.

By direct computation, we have that

$$[(f_1,g_1,h_1),(f_2,g_2,h_2)] = (0,0,f_1g_2 - f_2g_1), \qquad \text{(A.3)}$$

and thus $K(G) = \{(0,0,f_1g_2 - f_2g_1) \mid f_1,f_2 \in k[x],\ g_1,g_2 \in k[y]\}$. Define $H = \{(0,0,h) \mid h \in k[x,y]\}$. From (A.1) and (A.2) we deduce that $M \leqslant G$, and since $K(G) \subseteq H$, then $G' = \langle K(G) \rangle \leqslant H$. For the other inclusion, write $h(x,y) = \sum_{i,j} a_{i,j}x^iy^j$. By direct computation, we get that

$$\prod_{i,j}[(a_{i,j}x^i,0,0),(0,y^j,0)] = (0,0,h),$$

and thus, $H = G'$.

(ii) Apply (A.3).

[1] A group of this type is called a Heisenberg group.

(iii) Suppose that $(0,0,h) \in G'$ is a commutator. Therefore, by part ii) there exist $f_1(x) = \sum_i b_i x^i$ and $f_2(x) = \sum_i c_i x^i$ such that

$$h(x,y) = x^2 + xy + y^2 = \sum_i b_i x^i g_2(y) - \sum_i c_i x^i g_1(y). \tag{A.4}$$

In particular,

$$\begin{aligned} y^2 &= b_0 g_2(y) - c_0 g_1(y), \\ y &= b_1 g_2(y) - c_1 g_1(y), \\ 1 &= b_2 g_2(y) - c_2 g_1(y). \end{aligned}$$

Regarding $k[y]$ as a k-vector space, we have that the linearly independent set $\{1, y, y^2\}$ is contained in the subgroup generated by g_1 and g_2, and this is a contradiction.

(iv) If $k = \mathbb{F}_p$, $k[x,y]$ is replaced by $k[x,y]/(x^3, y^3, x^2y, xy^2)$, $k[x]$ by $k[x]/(x^3)$ and $k[y]/(y^3)$, then the correspoding group G has order $p^{12} = p^6 p^3 p^3$. It is clear that in this context we can apply parts i), ii) and iii). Hence, for every p prime number there is a group of order p^{12} such that $G' \neq K(G)$.

□

Exercise A.2.2. Check in GAP that the smallest group in which the property $K(G) \neq G'$ holds has order 96.

Solution. The following code solves the exercise:

```
CommutatorProp := function(G)
local commutators;
commutators := Set(List(Cartesian(G, G), x -> Comm(x)));
return Size(commutators) = Order(DerivedSubgroup(G));
end;
empty := AllSmallGroups([1..95], CommutatorProp, false);
examples := AllSmallGroups(96, CommutatorProp, false);
```

Since the list `empty` is empty and `examples` is not, the exercise is solved.

□

Appendix B

Source code

In this appendix we present all the programs that the author has written in the GAP [GAP22] system for this dissertation. The files are ordered in alphabetical order. At the beginning of each file the user can find the documentation of each program and after each file we show examples of the execution of each program. We finally note that all these files can be found also in the GitHub repository [Jua23], where is also available a README file which explains how to run these programs on one's computer.

AsCommutator.g

```
################################################################################
##
##  This file contains an implementation of an algorithm that writes any even
##  permutation as a commutator of two even permutations.
##
##  Created by Xabier de Juan Soriano on 2022
##  This file is part of the author's final degree dissertation.
##

DeclareGlobalFunction("CyclesToList");
DeclareGlobalFunction("CommutatorEvenPair");
DeclareGlobalFunction("CommutatorOddCycle");
DeclareGlobalFunction("AsCommutatorLists");

################################################################################
##
#M  AsCommutator( permutation )
##
##  This is the function that the user must call in order to write any even
##  permutation as a commutator of two even permutations.
##
##  input:
##      permutation: an even permutation, $r \in A_n$ (the value of n is assumed
##                      to be the smallest such thatr \in A_n)
##      If the permutation is not even an error message is raised.
##
##  output:
##      <list>[1]: even permutation $\phi \in A_n$
##      <list>[2]: even permutation $\psi \in A_n$
##      r = [\phi,\psi]
##
AsCommutator := function(permutation)
    local list;
    if SignPerm(permutation) = -1 then Error("permutation must be an even
        permutation"); fi;
    list := CyclesToList(permutation);
    return AsCommutatorLists(list[1],list[2],list[3]);
end;

################################################################################
##
##  AsCommutatorLists( cycles, cycles2, cycles3 )
##
##  This is function is not supposed to be used by the user
##
##  input:
##      cycles  : a list containing cycles of odd order
##      cycles2 : a list containing pairs of cycles of even order
##      cycles3 : a list containing 3-cycles
##          The cycle (a,b,c) corresponds to the list [ a , b , c ]
##          in this setting.
##
##  output:
##      The same as the main function
##
InstallGlobalFunction(AsCommutatorLists, function(cycles, cycles2, cycles3)
    local a, b, phi, psi, c, l, l1, l2, m, k, aux, max;

    # the particular cases
    aux := 0; # we will use this later
    # when we only have a cycle of order 3
    if  Length(cycles) = 0 and Length(cycles2) = 0 and Length(cycles3) = 1 then
        a := cycles3[1];
        max := Maximum(a);
        if max < 5 then max := 5; fi;
        b := [1..max];
        RemoveSet(b, a[1]);
        RemoveSet(b, a[2]);
        RemoveSet(b, a[3]);
        a := Concatenation(a, [b[Length(b)-1], b[Length(b)]]);
        return [(a[1] ,a[5], a[2]),(a[3], a[4], a[2])];
    fi;
    # when we only have a pair of a 2-cycle and a 4-cycle
    if Length(cycles) = 0 and Length(cycles3) = 0 and Length(cycles2) = 1 then
        # we know that Length(cycles2[1][1]) <= Length(cycles2[1][2])
        # this is because we are applying the function "CylesToList"
        if Length(cycles2[1][1]) = 2 and Length(cycles2[1][2]) = 4 then
            a := Concatenation(cycles2[1][2], cycles2[1][1]);
            return [(a[1],a[6],a[2],a[4],a[3]), (a[1],a[2],a[5],a[4],a[6])];
        fi;
    fi;
    # when we only have a pair of 2-cycles
    if Length(cycles) = 0 and Length(cycles3) = 0 and Length(cycles2) = 1 then
        if Length(cycles2[1][1]) = 2 and Length(cycles2[1][2]) = 2 then
            a := Concatenation(cycles2[1][1], cycles2[1][2]);
            return [ (a[1],a[3])(a[2],a[4]) , (a[1],a[4],a[3])  ];
        fi;
    fi;

    # the general case
    phi := ();
    psi := ();
    # the output will be: [phi,psi]
    for c in cycles do
        l := CommutatorOddCycle(c);
        phi := phi*l[1];
        psi := psi*l[2];
    od;
    # divide cases, if there are zero 3-cycles or >=2
    if Length(cycles3) <> 1 then
        for c in cycles2 do
            l := CommutatorEvenPair(c[1],c[2]);
            phi := phi*l[1];
            psi := psi*l[2];
        od;
        for c in cycles3 do
            l := CommutatorOddCycle(c);
            phi := phi*l[1];
            psi := psi*l[2];
        od;
    else # if there is only one 3-cycle
        if cycles <> [] then
            for c in cycles2 do
                l := CommutatorEvenPair(c[1],c[2]);
                phi := phi*l[1];
                psi := psi*l[2];
            od;
            l := CommutatorOddCycle(cycles3[1]);
            phi := phi*l[1];
            psi := psi*l[2];
            # from here we jump to "if SignPerm(phi) = -1 then"
        elif cycles2 <> [] then # redundant?
            for c in cycles2 do
                a := c[1];
                b := c[2];
                k := Length(a)/2;
                m := Length(b)/2 + k;
                if m >= 4 then
                    for c in cycles2 do
                        l := CommutatorEvenPair(c[1],c[2]);
                        phi := phi*l[1];
                        psi := psi*l[2];
                    od;
                    l := CommutatorOddCycle(cycles3[1]);
                    phi := phi*l[1];
                    psi := psi*l[2];
                    aux := 1;
                    break;
                    # from here we jump to "if SignPerm(phi) = -1 then"
                fi;
            od;
            # if we end up here that means that we are in the situation of the
                final part of the proof
            if aux = 0 then
                c := cycles3[1];
                a := cycles2[1][1];
                b := cycles2[1][2];
                if Length(b) = 2 then #2x2
                    a := Concatenation(c,a,b);
                    l1 := [a[4], a[5], a[6]];
                    l2 := [a[4], a[7], a[6]];
                    phi := phi * (a[1],a[2]) * MappingPermListList(l2,Reversed(l1
                        ));
                    psi := psi * (a[1],a[3],a[2]) * CycleFromList(l2);
                    if SignPerm(phi) = -1 then
                        phi := (a[1],a[4])*(a[3],a[7])*(a[2],a[6])*phi;
                    fi;
                else # 2x4
                    a := Concatenation(c,a,b);
                    phi := phi*(a[1],a[2] ) * (a[4], a[8], a[6], a[5], a[9], a
                        [7]);
                    psi := psi*(a[1], a[3], a[2]) *( a[4], a[5], a[9], a[8], a
                        [6]);
                fi;
                # compute the rest in any case
                for c in cycles2{[2..Length(cycles2)]} do
                        l := CommutatorEvenPair(c[1],c[2]);
                        phi := phi*l[1];
                        psi := psi*l[2];
                od;
            fi;
        fi;
    fi;

    #in order to ensure that phi \in An
    if SignPerm(phi) = -1 then
        # if there is some odd cycle it is easy to ensure phi \in An
        # we wil assume that the number of 3-cycles != 1
        if cycles <> [] then
            a := cycles[1];
            m := QuoInt(Order(CycleFromList(a)),2);
            # as it is said in the proof of the thm
            if m mod 2 = 0 then
                return [ (a[2],a[3])*phi, psi];
            else
                return [ (a[3],a[4])*phi, psi];
            fi;
        fi;
        # if there is some pair of even permutations to whom we have applied the
            lemma is easy to see phi \in An
        if cycles2 <> [] then
            for c in cycles2 do
                a := c[1];
                b := c[2];
                k := Length(a)/2;
                m := Length(b)/2 + k;
                if m >= 4 then
                    a := Concatenation(a,b);
                    # as in the proof
                    if m mod 2 = 0 then
                        return [(a[2],a[m])*phi, psi];
                    else
                        return [(a[4],a[m+1])*phi, psi];
                    fi;
                fi;
            od;
        fi;
        # if there is an odd number of 3-cycles
        if Length(cycles3) mod 2 = 1 then
            a := Concatenation(cycles3[1],cycles3[2]);
            return [(a[1],a[4])*(a[3],a[6])*(a[2],a[5])*phi,psi];
        fi;
    fi;
    return [phi, psi];
end);
```

```

#############################################################################
##
##  CommutatorOddCycle( cycle )
##
##  "cycle" is a list
##  Writes a cycle of odd length as a product of two cycles of odd length
##
InstallGlobalFunction(CommutatorOddCycle, function(cycle)
    # cycles is a list
    local m, l1, l2, phi;
    m := QuoInt(Order(CycleFromList(cycle)),2);
    if m mod 2 = 0 then
        l1 := cycle{[1..m+1]};
        l2 := Concatenation([cycle[1]], cycle{[m+2..2*m+1]});
    elif m = 1 then
        return [ (cycle[1],cycle[2]) , (cycle[1],cycle[3],cycle[2])  ];
    else # m odd and m>=5
        l1 := Concatenation([cycle[1]], [cycle[m+2]], cycle{[2..m+1]});
        l2 := Concatenation([cycle[1]], [cycle[m+2]], [cycle[2]], cycle{[m+3..2*m
            +1]});
    fi;
    phi := MappingPermListList(l2, Reversed(l1));
    return [phi, CycleFromList(l2)];
end);

#############################################################################
##
##  CommutatorEvenPair( t1, t2 )
##
##  t1 and t2 are lists. It writes a product of two cycles of even length as a
##  product of two cycles of odd length
##
InstallGlobalFunction(CommutatorEvenPair, function(t1, t2)
    # t1 and t2 are lists
    local k, m, t, cycle, l1, l2, phi;
    # we can suppose that Length(t1) <= Length(t2)
    k := Length(t1)/2;
    m := Length(t2)/2 + k;
    cycle := Concatenation(t1, t2);
    # some particular cases
    if m = 3 then
        cycle := Concatenation(t2,t1);
        return [(cycle[1],cycle[6],cycle[2],cycle[4],cycle[3]),(cycle[1],cycle
            [2],cycle[5],cycle[4],cycle[6])];
    elif m = 2 then
        return [ (cycle[1], cycle[3])*(cycle[2], cycle[4]) , (cycle[1], cycle[4],
             cycle[3])];
    fi;
    # the general case
    if m mod 2 = 0 then
        l1 := cycle{[1..m+1]};
        l2 := Concatenation([cycle[1]], cycle{[m+2..2*m]},[cycle[2*k+1]]);
    else
        l1 := Concatenation([cycle[1], cycle[m+2]], cycle{[2..m+1]});
        l2 := Concatenation([cycle[1], cycle[m+2], cycle[2]], cycle{[m+3..2*m]},[
            cycle[2*k+1]]);
    fi;
    phi := MappingPermListList(l2, Reversed(l1));
    return [phi, CycleFromList(l2)];
end);

```

```
InstallGlobalFunction(CyclesToList, function(permutation)
    # outputs a list with the disjoint cycles of "permutation"
    local cycles, cycles2, cycles22, cycles3, lista, l, j, c, e, N;
    N := Length(ListPerm(permutation));
    cycles := [];
    cycles2 := [];
    cycles22 := [];
    cycles3 := [];
    l := [1..N];
    while l <> [] do
        c := Cycle(permutation,l[1]);
        for e in c do
            RemoveSet(l,e);
        od;
        # cycles of order dont appear in the disjoint cycle decomposition
        if Length(c) >= 2 then
            # we want to divide the cycles of odd order, even order and 3-cycles
            if Length(c) mod 2 = 0 then # cycles of even order
                cycles2 := Concatenation(cycles2,[c]);
            elif Length(c) = 3 then # 3-cycles
                cycles3 := Concatenation(cycles3, [c]);
            else # cycles of odd order
                cycles := Concatenation(cycles,[c]);
            fi;
        fi;
    od;
    # write in pairs the cycles of even order
    if Length(cycles2) mod 2 = 0 then
        for e in [1,3..Length(cycles2)-1] do
            # in order to ensure that the one with smallest order appears in the
                left
            if Length(cycles2[e]) > Length(cycles2[e+1]) then
                cycles22 := Concatenation(cycles22,[ [cycles2[e+1], cycles2[e] ]
                    ]);
            else
                cycles22 := Concatenation(cycles22,[ [cycles2[e], cycles2[e+1] ]
                    ]);
            fi;
        od;
    fi;
    return [cycles, cycles22, cycles3];
end);
```

Example B.0.1. We take a random permutation in A_{21} and write it as a commutator. We check that the function works correctly.

```
gap> g := (1,15)(2,5)(3,19,8,12,4,10,7,20,16,9)(6,13,17)(11,21,18,14);;
gap> AsCommutator(g);
[ (1,2)(3,11,12,10)(4,8,7,14,9)(5,15)(6,13)(16,21,19,20,18),
    (1,5,2)(3,11,4,21,10,7,20,16,9)(6,17,13) ]
gap> g = Comm(AsCommutator(g));
true
gap> SignPerm(AsCommutator(g)[1]); SignPerm(AsCommutator(g)[2]);
1
1
```

CharTab.g

```
################################################################################
##
##  This file contains an implementation of an algorithm which computes computes
##  the table of irreducible characters of any finite group.
##
##  Created by Xabier de Juan Soriano on 2023
##  This file is part of the author's final degree dissertation.
##

DeclareGlobalFunction("NewPrint");
DeclareGlobalFunction("FieldToMod");
DeclareGlobalFunction("BestMat");
DeclareGlobalFunction("ClassMap");
DeclareGlobalFunction("ClassMatrixColumn");
DeclareGlobalFunction("ClassMatrix");

################################################################################
##
#M  CharTab( G, permRepr[, mode] )
##
##  Computes the table of irreducible characters of G using a simplified version
##  of the Burnside/Dixon/Schneider algorihm.
##
##  input:
##      G              : finite group.
##      permRepr       : if equal to true, a permutation representation of G is
##                          computed so all the computations are performed in
##                          this permutation group.
##                          Otherwise, the computations are performed in a matrix
##                          group or in the structure that the given group has.
##      mode(optional): when mode="silent" the program doesn't print any text
##                          regarding the progress of the computation.
##                          Omiting this argument can be helpful when dealing
##                          with large groups as it can help the user to
##                          estimate how much time of computation is left.
##
##  output:
##      <list>[1]: table of characters of G as a matrix
##                 (the rows are sorted by the degree)
##      <list>[2]: representatives of the conjugacy classes of G in the same
##                 order as the columns of the character table.
##
##  remarks: if the user is only interested in the values of the character
##              table, it is recommended to call the function with
##              true as a second argument, since the computations
##              will be performed faster in that case.
##
CharTab := function(G, permRepr, mode...)
    local CT, i, j, k, l, m, v, d2, x, sqrtt, Bv, K, A, s, V_i, V_, B_, b, h, D_,
         pol, I, q, prev;

    # we want permutation groups
    if permRepr = true and IsPermGroup(G) = false then G := Image(
        IsomorphismPermGroup(G)); fi;
    if IsPermGroup(G) then permRepr := true; fi;

    CT := rec();

    NewPrint(mode, "computing conjugacy classes...");
    CT.order := Size(G);
    CT.C := ConjugacyClasses(G);
    CT.CO := List(CT.C, c -> Size(c));
    CT.g := List(CT.C, c -> Representative(c));
    CT.r := Length(CT.C);
    CT.gO := List(CT.g, x -> Order(x));
    CT.e := Lcm(CT.gO);
    NewPrint(mode, "complete");
    NewPrint(mode, " " );

    CT.invClassMap := List([1..CT.r],j->ClassMap(CT.g[j]^-1, CT.C, permRepr));
    CT.powClassMap := List([1..CT.r],j->List([0..CT.gO[j]-1],l->ClassMap(CT.g[j]^
        l, CT.C, permRepr)));


    NewPrint(mode, "finding the prime number p...");
    CT.p := 2*Int(CT.order^0.5)+1;
    while IsPrimeInt(CT.p) = false or RemInt(CT.p-1,CT.e) <> 0 do
        CT.p := CT.p + 2;
    od;
    NewPrint(mode, "complete");
    NewPrint(mode, " ");

    NewPrint(mode, "computing the eigenvalues of the matrices M_j...");
    CT.M := List([1..CT.r], x-> IdentityMat(CT.r)-IdentityMat(CT.r));
    CT.TM := List([1..CT.r], x-> IdentityMat(CT.r)-IdentityMat(CT.r));
    CT.M[1] := IdentityMat(CT.r);
    j := PositionMinimum(List(CT.C{[2..CT.r]}, c -> Size(c))) + 1;

    CT.M[j] := ClassMatrix(CT.C, CT.g, j, CT.r, permRepr);
    CT.TM[j] := Z(CT.p)^0 * CT.M[j];

    CT.V := Eigenspaces(GF(CT.p), CT.TM[j]);
    CT.D := List(CT.V, v -> Dimension(v));
    CT.B := List(CT.V, v -> Basis(v));

    while CT.D <> List(CT.D, v -> 1) do
        NewPrint(mode, "-----");
        NewPrint(mode, "Dimension of the eigenspaces (we want a list full of 1s):
            ");
        NewPrint(mode, String(CT.D));
        i := Difference([1..Length(CT.D)],Positions(CT.D,1)); # list of the V_i
            with dim V_i > 1
        Bv := List(i, v -> BasisVectors(CT.B[v])); # bases of the V_i
        I := Length(i); # V_1,...,V_I
        s := List([1..I], v -> Length(Bv[v]));
        K := List([1..I], v -> [1..s[v]]); # the columns of M_j we need to
            compute
        for v in [1..I] do
            for m in [1..s[v]] do
                K[v,m] := PositionNot(Bv[v,m],0*Z(CT.p));
            od;
        od;
        j := BestMat(K,CT.g,CT.r,CT.CO,CT.C,permRepr);
        A := ShallowCopy(TransposedMat(CT.M[j]));
        for k in K do
            for l in k do
                if A[l] = List([1..CT.r],x->0) then
                    # we only compute the ones that have not been computed yet
                    # if a row is all 0 means that it has not been computed yet
                    A[l] := ClassMatrixColumn(CT.C, CT.g, j, CT.r, l, permRepr);
                fi;
            od;
        od;
        # we do not compute the whole matrix M_j, only the columns we need
        CT.M[j] := TransposedMat(A);
        # compute the eigenvalues of the action of TM[j] on V_i for each V_i
        prev:=0;
        for v in [1..I] do
            A := List([1..s[v]], x -> (Bv[v,x]*CT.M[j]){K[v]});
            V_i := Eigenspaces(GF(CT.p), A);
            h := Length(V_i);
            V_ := [1..h];
            for q in [1..h] do
                b := BasisVectors(Basis(V_i[q]));
                V_[q] := VectorSpace(GF(CT.p), List(b, x -> x*Bv[v]  ) );
            od;
            B_ := List(V_, q -> Basis(q));
            D_ := List(V_, q -> Dimension(q));
            # update the V, B and D vectors
            Remove(CT.V,i[v]+prev);
            CT.V := Concatenation(V_,CT.V);
            Remove(CT.B,i[v]+prev);
            CT.B := Concatenation(B_,CT.B);
            Remove(CT.D,i[v]+prev);
            CT.D := Concatenation(D_,CT.D);
            prev := prev + Length(D_)-1;
        od;
    od;
    NewPrint(mode, "complete");
    NewPrint(mode, " ");

    # convert the eigenvectors of F_p to integers mod p
    CT.B := List(CT.B, v -> FieldToMod(v[1]));
    # normalize the eigenvectors to ensure that the first component equals 1
    CT.B := List(CT.B, v -> PowerModInt(v[1],-1,CT.p) * v);
    NewPrint(mode, "computing Theta[X]");
    # compute Theta(X)
    CT.TX := IdentityMat(CT.r);
    for i in [1..CT.r] do
        d2 := CT.order * PowerModInt(Sum([1..CT.r],j -> Size(CT.C[j]) * CT.B[i,j]
             * CT.B[i, CT.invClassMap[j]] mod CT.p ), -1, CT.p);
        sqrtt:= RootMod(d2, CT.p);
        if 2*sqrtt >= CT.p then
            CT.TX[i] := AbsInt(sqrtt-CT.p)*CT.B[i] mod CT.p;
        else
            CT.TX[i] := sqrtt*CT.B[i] mod CT.p;
        fi;
    od;
    NewPrint(mode, "complete");

    NewPrint(mode, "recovering X from Theta X...");
    CT.omega := IntFFE(Z(CT.p)^((CT.p-1)/CT.e)); # element with multiplicative
        order e (as an integer < p)
    CT.m := List([1..CT.e], x -> IdentityMat(CT.r) );
    # pre-compute some values
    CT.powOrder := List([1..CT.r],j->PowerModInt(CT.gO[j],-1,CT.p));
    for i in [1..CT.r] do
        for j in [1..CT.r] do
            s := CT.gO[j]; # order of g[j]
            for k in [1..CT.e] do
                if (k-1) mod (CT.e/s) = 0 then
                    CT.m[i,j][k] := ( CT.powOrder[j] * Sum([0..s-1],l->
                        PowerModInt(CT.omega,-(k-1)*l,CT.p) * CT.TX[i, CT.
                        powClassMap[j,l+1]] mod CT.p  )) mod CT.p;
                else
```

```
                    CT.m[i,j][k] := 0;
                fi;
            od;
        od;
    od;
    NewPrint(mode, "complete");
    NewPrint(mode, "writing the character table...");
    # write the character table
    CT.X := IdentityMat(CT.r)-IdentityMat(CT.r);
    CT.Zeta := E(CT.e);
    for i in [1..CT.r] do
        for j in [1..CT.r] do
            #pol := List([1..CT.e], k-> CT.m[i,j][k]);
            CT.X[i,j] := ValuePol(CT.m[i,j], CT.Zeta);
        od;
    od;
    NewPrint(mode, "sorting the rows of the table");
    # sort the table by degrees
    SortBy(CT.X, x -> x[1]);
    # first character must be the trivial one
    j := Position(CT.X, List([1..CT.r],x->1));
    CT.X_ := ShallowCopy(CT.X);
    CT.X_[1] := CT.X[j];
    CT.X_[j] := CT.X[1];
    return [CT.X_, CT.g];
end;

################################################################################
##
##  ClassMatrix( C, g, j, r, permRepr )
##
##  Computes the class matrix M_j.
##
##  input:
##      C : conjugacy classes of G
##      g : representatives of the conj. classes
##          (it is assumed that it is given in the same order as C)
##      j : integer with 1<=j<=r
##      r : # of conj. classes
##
InstallGlobalFunction(ClassMatrix, function(C, g, j, r, permRepr)
    local l, M;
    M := IdentityMat(r);
    for l in [1..r] do
        M[l] := ClassMatrixColumn(C, g, j, r, l, permRepr);
    od;
    return TransposedMat(M);
end);

################################################################################
##
##  ClassMatrixColumn( C, g, j, r, l, permRepr )
##
##  Computes the l-th column of the class matrix M_j. It is returned as a row
##
InstallGlobalFunction(ClassMatrixColumn, function(C, g, j, r, l, permRepr)
    local x, y, z, k, c, v;
    z := g[l];
    v := List([1..r], x -> 0);
    # the first column is easy to compute
    if l = 1 then
        v[ClassMap(g[j]^-1, C, permRepr)] := Size(C[j]);
        return v;
    fi;
    for x in C[j] do
        y := x^-1*z;
        k := ClassMap(y,C,permRepr);
        v[k] := v[k] + 1;
    od;
    return v; # this is a row
end);

################################################################################
##
##  ClassMap( g, C, permRepr )
##
##  Computes the image of g under the class map
##  C is a list of all the conjugacy classes
##
InstallGlobalFunction(ClassMap, function(g, C, permRepr)
    local c, j, possible, x;
    if permRepr = true then
        possible := Filtered([1..Length(C)], x->CycleStructurePerm(Representative
            (C[x]))=CycleStructurePerm(g));
    else # this is why we prefer permutation groups
        possible := Filtered([1..Length(C)], x->Order(Representative(C[x]))=Order
            (g));
    fi;
    if Length(possible) = 1 then return possible[1]; fi;
    j := 1;
    while j < Length(possible) do
        if g in C[possible[j]] then
            return possible[j];
        else
            j := j+1;
        fi;
    od;
    return possible[j];
end);

################################################################################
##
##  BestMat( K, g, r, CO, C, permRepr )
##
##  Returns the value j such that computing the matrix M_j is the best option
##
InstallGlobalFunction(BestMat, function(K, g, r, CO, C, permRepr)
    local j, j_, k, best, val, columns;
    best := [1,0];
    columns := [];
    for j in [2..r] do
        columns := [];
        j_ := ClassMap(g[j]^-1, C, permRepr);
        val := 0;
        for k in K do
            UniteSet(columns,k);
            if Position(k,j_) <> fail then
                val := val + 1;
            fi;
        od;
        val := val/CO[j];
        val := val/Size(columns);
        if best[2] < val then
            best[2] := val;
            best[1] := j;
        fi;
    od;
    return best[1];
end);

InstallGlobalFunction(FieldToMod, function(V)
    local x;
    return List(V, x -> IntFFE(x));
end);

InstallGlobalFunction(NewPrint, function(mode, str)
    if mode = [] then
        Info(InfoWarning,1,str);
    fi;
end);
```

Example B.0.2. We compute the table of irreducible characters of A_6 and $\mathrm{SL}_2(4)$. The group A_6 is represented as a permutation group in GAP. On the other hand, $\mathrm{SL}_2(4)$ is represented as a matrix group.

```
gap> C:=CharTab(AlternatingGroup(6),false,"silent");; # in this case it is the
    same to write false or true in the second argument
gap> Display(C[1]);
[ [  1,  1,  1,  1,  1,              1,               1  ],
  [  5,  1, -1,  2, -1,              0,               0  ],
  [  5,  1,  2, -1, -1,              0,               0  ],
  [  8,  0, -1, -1,  0,   -E(5)-E(5)^4, -E(5)^2-E(5)^3  ],
  [  8,  0, -1, -1,  0, -E(5)^2-E(5)^3,   -E(5)-E(5)^4  ],
  [  9,  1,  0,  0,  1,             -1,              -1  ],
  [ 10, -2,  1,  1,  0,              0,               0  ] ]
gap> Print(C[2]);
[ (), (1,2)(3,4), (1,2,3), (1,2,3)(4,5,6), (1,2,3,4)(5,6), (1,2,3,4,5),
    (1,2,3,4,6) ]
gap> D := CharTab(SpecialLinearGroup(2,4),true,"silent");;
gap> Display(D[1]);
[ [  1,  1,  1,              1,              1 ],
  [  3, -1,  0, -E(5)^2-E(5)^3,   -E(5)-E(5)^4 ],
  [  3, -1,  0,   -E(5)-E(5)^4, -E(5)^2-E(5)^3 ],
  [  4,  0,  1,             -1,             -1 ],
  [  5,  1, -1,              0,              0 ] ]
gap> Print(D[2]);
[ (), ( 5, 6)( 7, 8)( 9,11)(10,12)(13,16)(14,15),
    ( 2, 3, 4)( 5,13, 9)( 6,15,12)( 7,16,10)( 8,14,11),
    ( 2, 5,10,11, 7)( 3, 9,15,16,12)( 4,13, 8, 6,14),
    ( 2, 5,14,16, 8)( 3, 9, 7, 6,10)( 4,13,12,11,15) ]
gap> F := CharTab(SpecialLinearGroup(2,4),false,"silent");;
# this character table is equivalent to the previous one
gap> Display(F[1]);
[ [  1,  1,              1,              1,  1  ],
  [  3, -1, -E(5)^2-E(5)^3,   -E(5)-E(5)^4,  0  ],
  [  3, -1,   -E(5)-E(5)^4, -E(5)^2-E(5)^3,  0  ],
  [  4,  0,             -1,             -1,  1  ],
  [  5,  1,              0,              0, -1  ] ]
# we get the elements represented in a different way
gap> Print(F[2]);
[    [ [ Z(2)^0, 0*Z(2) ], [ 0*Z(2), Z(2)^0 ] ],
    [ [ 0*Z(2), Z(2)^0 ], [ Z(2)^0, 0*Z(2) ] ],
    [ [ 0*Z(2), Z(2)^0 ], [ Z(2)^0, Z(2^2) ] ],
    [ [ 0*Z(2), Z(2)^0 ], [ Z(2)^0, Z(2^2)^2 ] ],
    [ [ Z(2^2), 0*Z(2) ], [ 0*Z(2), Z(2^2)^2 ] ] ]
```

OreTest1.g

```
################################################################################
##
##  This file contains an implementation of an algorithm that checks whether
##  in a group all elements are commutators
##
##  Created by Xabier de Juan Soriano on 2023
##  This file is part of the author's final degree dissertation.
##

################################################################################
##
##  OreTest1( CT )
##
##  input:
##      CT: ordinary character table of the group G as a matrix with values
##          in a cyclotomic field https://docs.gap-system.org/doc/ref/chap18.html
##
##  output:
##      returns true if all the elements of G are commutators and false otherwise
##
OreTest1 := function(CT)
    local i, j, r;
    r := Length(CT);
    CT := TransposedMat(CT);
    for i in [2..r] do
        if Sum([1..r], j->CT[i,j]/CT[1,j]) = 0 then
            return false;
        fi;
    od;
    return true;
end;
```

Example B.0.3. We check Ore's conjecture for A_6, S_6 and $\mathrm{SL}_2(8)$. In order to compute the character tables, we apply the previous function

```
gap> OreTest1(CharTab(AlternatingGroup(6),true,"silent")[1]);
true
gap> OreTest1(CharTab(SymmetricGroup(6),true,"silent")[1]);
false
gap> OreTest1(CharTab(SpecialLinearGroup(2,8),true,"silent")[1]);
true
```

OreTest2.g

```
################################################################################
##
##  This file contains an implementation of an algorithm that checks whether in
##  a group all elements are commutators
##
##  Created by Xabier de Juan Soriano on 2023
##  This file is part of the author's final degree dissertation.
##

################################################################################
##
##  OreTest2( G )
##      the program halts if and only if every element of G is a commutator
##
##   input:
##      G   : finite group G
##
##   output:
##      returns true if all the elements of G are commutators
##
OreTest2 := function(G)
    local g, x, repr, order, index_cent, new_repr;
    order := Order(G);
    # we want permutations groups
    if IsPermGroup(G) = false then G := Image(IsomorphismPermGroup(G)); fi;
    repr := [()]; # to store the different representative of the classes
    index_cent := 1; # cummulative sum of the index of the centralizers
    while order <> index_cent do
        new_repr := true;
        g := Comm(Random(G),Random(G));
        if g <> () then
            for x in repr do
            # first cheap conjugation test
            if CycleStructurePerm(x) = CycleStructurePerm(g) then
                if IsConjugate(G, x, g) then
                    new_repr := false;
                    break;
                fi;
            fi;
            od;
            if new_repr then
                Append(repr,[g]);
                index_cent := index_cent + order/Size(Centralizer(G, g));
            fi;
        fi;
    od;
    return true;
end;
```

Example B.0.4. We check Ore's conjecture for $PSL_3(3)$.

```
gap> OreTest2(ProjectiveSpecialLinearGroup(3,3));
true
```

Bibliography

[Bre22] T. Breuer. *GAP package CTblLib*. 2022. URL: https://www.gap-system.org/Packages/ctbllib.html.

[Bur55] W. Burnside. *Theory of Groups of Finite Order*. New York: Dover Publications, 1955.

[Cel+95] F. Celler et al. "Generating random elements of a finite group". In: *Communications in Algebra* 23.13 (1995), pp. 4931–4948. DOI: 10.1080/00927879508825509.

[Con20] K. Conrad. *Simplicity of* $\mathrm{PSL}_n(F)$. 2020. URL: https://kconrad.math.uconn.edu/blurbs/grouptheory/PSLnsimple.pdf (Retrieved 05/05/2023).

[CS97] J.J. Cannon and B. Souvignier. "On the Computation of Conjugacy Classes in Permutation Groups". In: *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation, ISSAC 1997, Maui, Hawaii, USA, July 21-23, 1997*. ACM, 1997, pp. 392–399. DOI: 10.1145/258726.258855.

[CS99] H. Cuypers and A. Steinbach. "Linear transvection groups and embedded polar spaces". In: *Inventiones Mathematicae* 137 (1999), pp. 169–198. DOI: 10.1007/s002220050328.

[Dir37] P.G.L. Dirichlet. "Beweis des Satzes, dass jede unbegrenzte arithmetische Progression, deren erstes Glied und Differenz ganze Zahlen ohne gemeinschaftlichen Factor sind, unendlich viele Primzahlen enthält". In: *Abhandlungen der Königlich Preußischen Akademie der Wissenschaften* 48 (1837), pp. 45–71.

[Dix67] J.D. Dixon. "High speed computation of group characters". In: *Numer. Math.* 10 (1967), pp. 446–450. DOI: 10.1007/BF02162877.

[EG98] E.W. Ellers and N.L. Gordeev. "On the Conjectures of J. Thompson and O. Ore". In: *Transactions of the American Mathematical Society* 350.9 (1998), pp. 3657–3671.

[GAP22] *GAP – Groups, Algorithms, and Programming, Version 4.12.0*. The GAP Group. 2022. URL: https://www.gap-system.org/.

[Gro01] L.C. Grove. *Classical Groups and Geometric Algebra*. American Mathematical Society, 2001.

[GT15] R.M. Guralnick and P.H. Tiep. *Effective Results on the Waring Problem for Finite Simple Groups*. 2015. arXiv: 1302.0333.

[HEO05] D.F. Holt, B. Eick, and E.A. O'Brien. *Handbook of Computational Group Theory*. Discrete Mathematics and Its Applications. CRC Press, 2005.

[Hul93] A. Hulpke. "Zur Berechnung von Charaktertafeln". Diploma thesis. Rheinisch Westfälische Technische Hochschule, 1993.

[Isa94] I.M. Isaacs. *Character Theory of Finite Groups*. Dover, 1994.

[Ito51] N. Ito. "A theorem on the alternating group $\mathfrak{A}_n$ $(n \geqslant 5)$". In: *Math. Japonicae* 2 (1951), pp. 59–60.

[Jor70] C. Jordan. *Traité des substitutions et des équations algébriques*. Gauthier-Villars, 1870.

[Jua23] X. de Juan. *TFGcode*. https://github.com/xdejs/TFGcode. 2023.

[KM05] L.C. Kappe and R. Morse. "On commutators in groups". In: *Journal of Group Theory* 8 (2005), pp. 415–429.

[Lan02] S. Lang. *Algebra*. Springer, 2002.

[LOST] M. Liebeck et al. "The Ore conjecture". In: *Journal of the European Mathematical Society* 012.4 (2010), pp. 939–1008. DOI: 10.4171/JEMS/220.

[Mal14] G. Malle. "The proof of Ore's conjecture [after Ellers-Gordeev and Liebeck-O'Brien-Shalev-Tiep]". In: *Séminaire Bourbaki volume 2012/2013 : exposés 1059-1073 - Avec table par noms d'auteurs de 1948/49 à 2012/13*. Astérisque 361. talk:1069. Société mathématique de France, 2014.

[Mil99] G.A. Miller. "On the commutators of a given group". In: *Bulletin of the American Mathematical Society* 6.3 (1899), pp. 105–109.

[NPC84] J. Neubüser, H. Pahlings, and E. Cleuvers. "Each sporadic finasig G has a class C such that $CC = G$". In: *Abstracts AMS*. Vol. 34. 6. 1984.

[Ore51] O. Ore. "Some Remarks on Commutators". In: *Proceedings of the American Mathematical Society* 2.2 (1951), pp. 307–314. DOI: 10.2307/2032506.

[Rot95] J.J. Rotman. *An Introduction to the Theory of Groups*. Springer New York, 1995.

[Sch90] G. Schneider. “Dixon’s character table algorithm revisited”. In: *Journal of Symbolic Computation* 9.5 (1990), pp. 601–606. DOI: 10.1016/S0747-7171(08)80077-6.

[TL65] K’en-ch’eng Ts’eng and Chiung-sheng Li. “On the commutators of the simple Mathieu groups”. In: *J. China Univ. Sci. Techn.* 1 (1965), pp. 43–48.

[Wey39] H. Weyl. *The Classical Groups. Their Invariants and Representations*. Princeton University Press, 1939.

[Wil09] R.A. Wilson. *The Finite Simple Groups*. Springer, 2009.