2012 IEEE 27th Conference on Computational Complexity

Matrix Isomorphism of Matrix Lie Algebras

Joshua A. Grochow
Department of Computer Science
The University of Chicago
Chicago, IL, USA
Email: joshuag@cs.uchicago.edu

***Abstract*—We study the problem of** MATRIX ISOMORPHISM OF MATRIX LIE ALGEBRAS (MATISOLIE)**. Lie algebras arise centrally in areas as diverse as differential equations, particle physics, group theory, and the Mulmuley–Sohoni Geometric Complexity Theory program. A matrix Lie algebra is a set L of matrices that is closed under linear combinations and the operation [A,B] = AB - BA. Two matrix Lie algebras L, L' are matrix isomorphic if there is an invertible matrix M such that conjugating every matrix in L by M yields the set L'.**

We show that certain cases of MATISOLIE**—for the wide and widely studied classes of *semisimple* and *abelian* Lie algebras—are equivalent to** GRAPH ISOMORPHISM **and** LINEAR CODE EQUIVALENCE**, respectively. On the other hand, we give polynomial-time algorithms for other cases of** MATISOLIE**, which allow us to mostly derandomize a recent result of Kayal on affine equivalence of polynomials.**

***Keywords*-Lie algebras, graph isomorphism, determinant, linear code equivalence, affine equivalence of polynomials, algorithms**

I. INTRODUCTION

A *matrix Lie algebra* is any set of $n \times n$ matrices closed under the following operations: multiplication by scalars, matrix addition, and a multiplication-like operation denoted $[A, B] := AB - BA$. Lie algebras are an important tool in areas as diverse as differential equations [1], [2], particle physics [3], group theory [4], [5], [6], and the Mulmuley–Sohoni Geometric Complexity Theory program [7].

In complexity theory, Kayal [8] has recently used Lie algebras in AFFINE EQUIVALENCE, which arises in many areas of complexity: factoring integers, permanent versus determinant, matrix multiplication, lower bounds for depth-three circuits, and several more (see [8, §1.1]). Kayal essentially used a special case of MATISOLIE to give a randomized polynomial-time algorithm to decide when a function can be gotten from the determinant by an invertible linear change of variables. This is the affine equivalence problem for the determinant.

The following are examples of Lie algebras, which should help give their flavor, and introduces some of those Lie algebras on which we prove results, namely abelian, diagonalizable, and semisimple:

1) The collection of all $n \times n$ matrices.
2) The collection of all diagonal $n \times n$ matrices is a Lie algebra of dimension n. Any two diagonal matrices D_1, D_2 commute. Since $D_1 D_2 - D_2 D_1 = 0$ this is a Lie algebra. Any Lie algebra in which all matrices commute is called *abelian*.
3) In fact, *any* collection of diagonal matrices that is closed under taking linear combinations is a Lie algebra, for the same reason as above. Furthermore, if $\mathcal{D}$ is such a Lie algebra, then $A\mathcal{D}A^{-1}$ is as well, since conjugating by A preserves the fact that all the matrices in $\mathcal{D}$ commute. Any Lie algebra conjugate to a set of diagonal matrices is called *diagonalizable*.
4) The collection of all $n \times n$ matrices with trace zero. Since $\mathrm{tr}(AB - BA) = 0$ for any A, B, this is also a Lie algebra. This is an example of a *simple* Lie algebra.
5) The collection of all $2n \times 2n$ matrices of the form $\begin{pmatrix} C & 0 \\ 0 & D \end{pmatrix}$ where C, D are $n \times n$ matrices and $\mathrm{tr}\, C + \mathrm{tr}\, D = 0$.

Two matrix Lie algebras $\mathcal{L}_1, \mathcal{L}_2$ are *matrix isomorphic* if there is an invertible matrix A such that $\mathcal{L}_1 = A\mathcal{L}_2 A^{-1}$, that is, for every matrix M in $\mathcal{L}_2$, AMA^{-1} is in $\mathcal{L}_1$, and $\mathcal{L}_1$ and $\mathcal{L}_2$ have the same dimension. Although it was not phrased this way, Kayal gave a randomized reduction from AFFINE EQUIVALENCE for the determinant to MATISOLIE for Lie algebras abstractly isomorphic to example (5) (see Section III for the difference between abstract and matrix isomorphism). However, where he uses properties very specific to the Lie algebras associated to permanent and determinant that can be computed using randomization, we are able to instead use a deterministic approach to the more general problem of MATISOLIE.

A. Results

We show that certain cases of MATISOLIE are solvable in polynomial time. We also show that extending these cases is difficult, as such an extension is equivalent to GRAPH ISOMORPHISM in one case and at least as hard as GRAPH ISOMORPHISM in the other case. One of these cases is strong enough to mostly derandomize Kayal's result [8] on testing affine equivalence to the determinant (see Section VI).

We now give the formal definition of MATISOLIE. Since Lie algebras are closed under taking linear combinations, we can give them as input to algorithms by providing a linear basis.

1093-0159/12 $26.00 © 2012 IEEE
DOI 10.1109/CCC.2012.34

Problem: MATRIX ISOMORPHISM OF MATRIX LIE ALGEBRAS (MATISOLIE)
Input: Two Lie algebras $\mathcal{L}_1$ and $\mathcal{L}_2$ of $n \times n$ matrices, given by basis elements.
Output: An invertible $n \times n$ matrix A such that $A\mathcal{L}_1 A^{-1} = \mathcal{L}_2$, if such A exists, otherwise "the Lie algebras are not matrix isomorphic."

Recall the definitions of abelian and diagonalizable from examples (2) and (3) above, respectively.

Corollary II.2. ABELIAN DIAGONALIZABLE MATISOLIE *of $n \times n$ matrices can be solved by a $poly(n)$-time f-algorithm*[1] *when the Lie algebras have dimension $O(1)$.*

Abelian Lie algebras are one of two fundamental building blocks of all Lie algebras. The other fundamental building blocks are the semisimple Lie algebras. A Lie algebra is semisimple if it is a direct sum of simple Lie algebras. Example (4) above is a simple Lie algebra; see Section III-C for definitions, and the full version [9, Chapter 4] for what we mean by "building blocks." For this other building block, we show a similar result:

Theorem IV.6. SEMISIMPLE MATISOLIE *of $n \times n$ matrices can be solved by a $poly(n)$-time f-algorithm when the Lie algebras have only $O(\log n)$ simple direct summands.*

Despite the $O(\log n)$ restriction, Theorem IV.6 is already strong enough to mostly derandomize Kayal's result (Corollary VI.1 below). Also, note that even a single simple Lie algebra can have unbounded dimension, as in example (4), let alone a semisimple one with $O(\log n)$ simple summands. Theorem IV.7 gives another class on which MATISOLIE is solvable in polynomial time.

For both results above, we show that removing the quantitative restrictions is likely to be difficult:

Theorem II.1. GRAPH ISOMORPHISM *polynomial-time many-one reduces to* ABELIAN DIAGONALIZABLE MATISOLIE*, when the Lie algebras may have unbounded dimension.*

Theorem IV.1. GRAPH ISOMORPHISM *is f-equivalent to* SEMISIMPLE MATISOLIE*, when the Lie algebras may contain an unbounded number of simple direct summands.*

In fact, we show that ABELIAN DIAGONALIABLE MATISOLIE is equivalent to CODE EQUIVALENCE. The code equivalence problem is to test whether two subspaces of a vector space can be made equal by permuting their coordinates. For codes over $\mathbb{F}_2$, CODE EQUIVALENCE was known to be as hard as GRAPH ISOMORPHISM [10]; we extend their proof to show that CODE EQUIVALENCE over *any field* is as hard as GRAPH ISOMORPHISM.

Finally, we combine the abelian and semisimple cases together, to show results on MATISOLIE when the Lie algebras are the direct sum of an abelian Lie algebra and a semisimple one:

Theorem V.2. MATISOLIE *for Lie algebras of $n \times n$ matrices can be determined by a $poly(n)$-time f-algorithm when the Lie algebras are a direct sum of an $O(1)$-dimensional abelian diagonalizable Lie algebra and a semisimple Lie algebra with $O(\log n)$ simple direct summands.*

Since abelian is a special case of abelian-plus-semisimple, this more general case is obviously as hard as CODE EQUIVALENCE when we drop the quantitative restrictions of the above theorem.

B. A note on the field and model of computation

For clarity we state all our results over $\mathbb{C}$ unless mentioned otherwise. However, all of our main theorems essentially (see the next section) hold over any algebraically closed field of characteristic zero, or finite fields of sufficiently large characteristic, depending on d and n, when the input consists of d-dimensional Lie algebras of $n \times n$ matrices. However, unlike so many cases in algorithms and complexity, the extension to finite fields requires more results than merely the Schwarz–Zippel Lemma; specifically, we need known results on the structure of Lie algebras in positive characteristic [11], [12], [13].

For our results to hold in the generality above, "semisimple" should everywhere be replaced by "classical," and "simple" should be replaced by "simple classical." In characteristic zero these terms are equivalent, but in positive characteristic they are not and the change is necessary. See the full version of the paper [9, Chapter 4] for details.

Many of the Lie-theoretic results needed for our results hold of any perfect field, with a few exceptions in characteristics $2, 3$. A field is *perfect* if either it has characteristic 0, or it has characteristic p and every element is a p-th power. In particular, finite fields are perfect. However, some of the algorithmic preliminaries we needed on Lie algebras have only been developed in characteristic zero or over finite fields. Infinite perfect fields of positive characteristic have often been overlooked, possibly because authors were concerned only with the bit-wise model of computation. In this paper we are primarily concerned with algebraic complexity—P-uniform arithmetic circuits, or equivalently the Blum–Shub–Smale (BSS) model [14] without constants—though essentially all of our results can be shown to hold when complexity is counted in terms of bits, whenever that makes sense—for example over $\mathbb{Q}$, the "rational complex numbers" $\mathbb{Q}(i)$, and finite fields.

C. A note on diagonalizing matrices

The reductions *from* GRAPH ISOMORPHISM and CODE EQUIVALENCE to various subcases of MATISOLIE require no additional assumptions. However, for the reductions in

[1] See Section I-C.

the opposite direction we diagonalize matrices as a subroutine. Diagonalizing a matrix is equivalent to factoring polynomials, under polynomial-time many-one (=Karp) reductions over $\mathbb{F}$ using, say, the BSS model.

In the settings in which we diagonalize matrices, we always assume that the eigenvalues lie in the ground field $\mathbb{F}$. In particular, "diagonalizable" in this paper means all eigenvalues are in $\mathbb{F}$. For algebraically closed fields this holds trivially, but for other fields we must add it explicitly. This assumption can be weakened to, for example, a polynomial-degree extension field of $\mathbb{F}$. Some such assumption seems necessary in order to use polynomial factorization, as the roots of a generic polynomial over $\mathbb{F}$ of degree n require a degree $n!$ extension of $\mathbb{F}$.

In this paper, an *f-algorithm* over $\mathbb{F}$ is an arithmetic algorithm over $\mathbb{F}$ with an oracle for finding roots of polynomials over $\mathbb{F}$. If the k coefficients of a polynomial have been written down, in k steps the coefficients may be replaced by the roots. We similarly define *f-reductions*. We denote (arithmetic) Karp reductions by $\leq_m$ and (arithmetic) Karp f-reductions by $\leq_m^f$. We use "f-time" to denote the running time of an f-algorithm.

The complexity of polynomial root-finding depends on the ground field, and is:

- Over $\mathbb{F}_q$, $poly(n, q)$ deterministic time [15], or $poly(n, \log q)$ Las Vegas randomized time [16]. The latter is polynomial in the bit-size.
- Over $\mathbb{C}$, NC approximation algorithms exist in both the algebraic and the Boolean models [17], but the exact complexity of our f-algorithms over $\mathbb{C}$ remains unknown, due to the possibility of distinct roots that are exponentially close together. We suspect that the properties of matrix Lie algebras can be used to avoid this pitfall, but have so far been unable to prove this. See the full paper for more details.

See the survey by Kaltofen [18] for more.

We thank an anonymous referee for pointing out the complexity of polynomial factorization. This issue is also present but not discussed in Kayal [8]. Kayal's results over $\mathbb{C}$ may suffer from the same issue mentioned above; over sufficiently large finite fields his results hold as stated, since all of his algorithms use randomness anyways.

D. Outline

In Section II we prove Theorems II.2 and II.1 on ABELIAN MATISOLIE; this section can be understood without any background on Lie algebras. Our other results require more knowledge of Lie algebras; we collect the necessary background in Section III. In Section IV we prove Theorems IV.1, IV.6, and IV.7 on SEMISIMPLE MATISOLIE. In Section V we prove our results on direct sums of abelian and semisimple Lie algebras, including Theorem V.2. In Section VI we show how to use the above machinery to mostly derandomize Kayal's result on testing affine equivalence to the determinant. In the final section, we discuss how close the abelian-plus-semisimple case is to the general case, directions toward the general case, and other future work, including potential ways to solve important special cases of the affine equivalence problem efficiently without having to efficiently solve GRAPH ISOMORPHISM.

II. ABELIAN DIAGONALIZABLE MATISOLIE AND CODE EQUIVALENCE

In this section we show that ABELIAN DIAGONALIZABLE MATISOLIE is f-Karp-equivalent to CODE EQUIVALENCE, and hence is at least as hard as GRAPHISO. The reduction to CODE EQUIVALENCE allows us to solve ABELIAN DIAGONALIZABLE MATISOLIE for constant-dimensional Lie algebras in polynomial f-time. As diagonalizable implies abelian, we hereafter refer to this as the diagonalizable case.

A d-dimensional *code* of length n over a field $\mathbb{F}$ is a d-dimensional subspace of $\mathbb{F}^n$. Codes are represented algorithmically by giving bases for them as subspaces. The symmetric group S_n acts on $\mathbb{F}^n$ by permutation of coordinates: for $\pi \in S_n$ and $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$, $\pi \cdot \vec{\alpha} = (\alpha_{\pi(1)}, \ldots, \alpha_{\pi(n)})$. S_n then acts on a subspace $V \subseteq \mathbb{F}^n$ by $\pi \cdot V = \{\pi \cdot v : v \in V\}$. The *code equivalence problem* is: given two codes C_1, C_2, determine whether there is a permutation $\pi \in S_n$ such that $\pi \cdot C_1 = C_2$.

Theorem II.1. DIAGONALIZABLE MATISOLIE $\leq_m$ CODE EQUIVALENCE $\leq_m^f$ DIAGONALIZABLE MATISOLIE. *The reductions show that* DIAGONALIZABLE MATISOLIE *for d-dimensional subspaces of $n \times n$ matrices is f-equivalent to* CODE EQUIVALENCE *for d-dimensional subspaces of $\mathbb{F}^n$, for any field $\mathbb{F}$.*

Before proving this theorem we give some of its consequences.

Corollary II.2. DIAGONALIZABLE MATISOLIE *of $n \times n$ matrices can be solved by a $poly(n)$-time f-algorithm when the Lie algebras have dimension $O(1)$.*

Proof: Babai (see [19, Theorem 7.1]) showed that, over any field $\mathbb{F}$, equivalence of d-dimensional linear codes of length n reduces to $\binom{n}{d}$ instances of $d \times (n-d)$ EDGE-COLORED BIPARTITE GRAPHISO. Each such instance can be solved in $poly(n) \cdot \min\{d!, (n-d)!\}$ time, so when $d = O(1)$ CODE EQUIVALENCE can be solved in polynomial time. By Theorem II.1, d-dimensional DIAGONALIZABLE MATISOLIE can be solved in polynomial f-time when $d = O(1)$. ∎

Corollary II.3. DIAGONALIZABLE MATISOLIE *over any field is* GRAPHISO*-hard.*

Proof: Petrank and Roth [10] showed that GRAPHISO Karp-reduces to CODE EQUIVALENCE over $\mathbb{F}_2$. Over an arbitrary field we use the same reduction, but an extension of their proof is required, which we give in Lemma II.4

below. Theorem II.1 then shows that DIAGONALIZABLE MATISOLIE is GRAPHISO-hard. ∎

Lemma II.4. GRAPH ISOMORPHISM *Karp-reduces to* LINEAR CODE EQUIVALENCE *over any field* $\mathbb{F}$.

Proof: Given a graph G, we construct the generator matrix for a code over $\mathbb{F}$ such that two graphs are isomorphic if and only if the codes are equivalent. Let $M(G) = [I_m|I_m|I_m|D]$ where $m = |E(G)|$, I_m is the $m \times m$ identity matrix, and D is the incidence matrix of G:

$$D_{e,v} = \begin{cases} 1 & \text{if } v \in e \\ 0 & \text{otherwise} \end{cases}$$

The *Hamming weight* of a vector over $\mathbb{F}$ is the number of non-zero entries. The following claim is essentially the crux of Petrank and Roth's argument, but generalized so as to apply over any field.

Claim: up to permutation and scaling of the rows, $M(G)$ is the unique generator matrix of its code which satisfies the following properties:

1) it is a $|E| \times (3|E| + |V|)$ generator matrix;
2) each row has Hamming weight ≤ 5;
3) any nondegenerate linear combination of two or more rows has Hamming weight ≥ 6

A linear combination of k rows is nondegenerate if all k of its coefficients are nonzero.

Proof of claim: First, $M(G)$ satisfies (1)–(3). The only part to check is (3): in the first $3m$ columns, any nondegenerate linear combination of $k \geq 2$ rows will have $3k \geq 6$ nonzero entries. Next, let C denote the code generated by the rows of $M(G)$. By (2) and (3) the rows of $M(G)$ are the *unique* vectors in C (up to scaling) of Hamming weight ≤ 5. Hence if M' is any other generator matrix of C satisfying (1)–(3), its rows must be scaled versions of the rows of $M(G)$ in some order. This proves the claim.

Now, suppose that $M(G_1)$ and $M(G_2)$ generate equivalent codes. Then there is a nonsingular matrix S and a permutation matrix P such that $M(G_1) = SM(G_2)P$. By the claim, $S = \Delta S'$ where Δ is diagonal and S' is a permutation matrix. However, since the first $3|E|$ columns of $M(G_1)$ and $M(G_2)$ only contain $0, 1$-entries, $\Delta = I$. The rest of the proof of the reduction, including the other direction, proceeds exactly as in Petrank and Roth [10]. ∎

Proof of Theorem II.1: Let $(A_1, \ldots, A_d), (B_1, \ldots, B_d)$ be two spanning sets of diagonalizable Lie algebras. Using root-finding and standard techniques in linear algebra, the A_i can be simultaneously diagonalized in polynomial time, so we may now assume that the A_i are in fact diagonal, rather than merely diagonalizable. Similarly for the B_i. Let $\mathcal{A}$, resp. $\mathcal{B}$, denote the Lie algebras spanned by the A_i, resp. B_i.

Claim: If $\mathcal{A}$ and $\mathcal{B}$ are diagonal, then they are matrix isomorphic if and only if they are matrix isomorphic via a permutation matrix.

By "flattening out" the entries of the diagonal matrices into "row" vectors the claim shows that d-dimensional MATISOLIE for *diagonal* Lie algebras of $n \times n$ matrices is Karp-equivalent to d-dimensional CODE EQUIVALENCE for codes of length n. The root-finding oracle is only needed for one direction to diagonalize the Lie algebras. Thus the claim will complete the proof of the theorem.

Proof of claim: Suppose $C\mathcal{A}C^{-1} = \mathcal{B}$. Since $\mathcal{A}$ and $\mathcal{B}$ are both diagonal, C must preserve the eigenspaces of every matrix in $\mathcal{A}$. The formalization of this notion will allow us to prove our claim. Let $\lambda_i \colon \mathcal{A} \to \mathbb{F}$ be the linear function $\lambda_i(A) = A_{ii}$. We can think of λ_i as a "simultaneous eigenvalue for the space $\mathcal{A}$ of matrices," generalizing the notion of an eigenvalue of a single matrix. Such functions are called *weights* in Lie theory, and they will play a significant role. Analogous to an eigenspace corresponding to an eigenvalue, there are *weight spaces* corresponding to weights. Namely, if $\lambda \colon \mathcal{A} \to \mathbb{F}$ is a weight, the corresponding weight space is

$$V_\lambda(\mathcal{A}) := \{v \in \mathbb{F}^n : Av = \lambda(A)v \text{ for all } A \in \mathcal{A}\}$$

It is these weight spaces that C must preserve in order for $C\mathcal{A}C^{-1}$ to be diagonal. For example, if every weight space is 1-dimensional—or equivalently, if for every pair of indices $1 \leq i < j \leq n$ there is some matrix $A \in \mathcal{A}$ with $A_{ii} \neq A_{jj}$—then C must be the product of a permutation matrix and a diagonal matrix. Since diagonal matrices commute, conjugating $\mathcal{A}$ by a diagonal matrix has no effect, and we may discard it; hence C may be taken to be a permutation matrix.

More generally, C may send $v \in V_{\lambda_1}$ into V_{λ_2} if and only if $CV_{\lambda_1} = V_{\lambda_2}$. Within each weight space, C may act in an arbitrary invertible manner. In other words, C is composed of invertible blocks of dimension $\dim V_{\lambda_i}$, the pattern in which these blocks appear is a permutation, and that permutation may send $i \mapsto j$ if and only if $\dim V_{\lambda_i} = \dim V_{\lambda_j}$. However, if C' has the same permutation pattern as C but all the blocks in C' are the identity, then $C\mathcal{A}C^{-1} = C'\mathcal{A}C'^{-1}$. Hence, without loss of generality, we may take C to be a permutation matrix, proving the claim. ∎

III. LIE ALGEBRA PRELIMINARIES

For the purposes of this paper, we highly recommend the book of de Graaf [11]. We summarize the necessary highlights here. For more general background on Lie algebras we recommend the standard books [4], [20]. For Lie algebras over non-algebraically closed fields or in positive characteristic, we recommend the books by Jacobson [20] and especially Seligman [12], though it is probably a good idea to have seen the case of Lie algebras over $\mathbb{C}$ before wading into modular Lie algebras [12].

A. Basic definitions

A *Lie algebra* is a vector space $\mathcal{L}$ together with a map $[\cdot, \cdot] \colon \mathcal{L} \times \mathcal{L} \to \mathcal{L}$, referred to as the *Lie bracket*, satisfying:

1) Skew-symmetry: $[v, v] = 0$ for all $v \in \mathcal{L}$ (equivalently in characteristic $\neq 2$, $[u, v] = -[v, u]$)
2) Bilinearity: $[\alpha v + \beta w, u] = \alpha[v, u] + \beta[w, u]$, and similarly for the second coordinate.
3) The Jacobi identity: $[u, [v, w]] + [w, [u, v]] + [v, [w, u]] = 0$. This is the "Lie algebra version" of associativity, and can be thought of as "the derivative of the associative law."

A *homomorphism* between Lie algebras $\mathcal{L}_1, \mathcal{L}_2$ is a linear map $\rho\colon \mathcal{L}_1 \to \mathcal{L}_2$ where $\rho([u, v]_{\mathcal{L}_1}) = [\rho(u), \rho(v)]_{\mathcal{L}_2}$. An *(abstract) isomorphism* is a bijective homomorphism; an *automorphism* is an isomorphism of $\mathcal{L}$ with itself.

A *matrix Lie algebra* is a set $\mathcal{L}$ of matrices where defining $[A, B] := AB - BA$ makes $\mathcal{L}$ a Lie algebra. In particular, the collection M_n of all $n \times n$ matrices is a matrix Lie algebra. Two matrix Lie algebras are *matrix isomorphic* if there is a matrix A such that $A\mathcal{L}_1 A^{-1} = \mathcal{L}_2$. A matrix isomorphism is a special case of an isomorphism since conjugating by A preserves the matrix Lie bracket: $A[M_1, M_2]A^{-1} = [AM_1A^{-1}, AM_2A^{-1}]$. When necessary, we specify "matrix isomorphism" versus "abstract isomorphism."

B. Describing Lie algebras as input to algorithms

An abstract Lie algebra is specified in algorithms by giving a basis for it as a vector space, say $v_1, \ldots, v_d$, and by its *structure constants* $c_{ij}^{(k)}$:

$$[v_i, v_j] = \sum_{k=1}^{n} c_{ij}^{(k)} v_k.$$

Because of the bilinearity of the bracket, the structure constants fully determine the bracket: $[\sum \alpha_i v_i, \sum \beta_j v_j] = \sum_{ijk} \alpha_i \beta_j c_{ij}^{(k)} v_k$. Each of the axioms of a Lie algebra translates into a condition on the structure constants, for example, skew-symmetry is equivalent to $c_{jj}^{(k)} = 0$ for all j, k.

Given a matrix Lie algebra $\mathcal{L} \subseteq M_n$, we may compute its structure constants to get an abstract isomorphic copy $\mathcal{L}'$ of $\mathcal{L}$ as follows. Suppose $\mathcal{L}$ is given by a basis of matrices $\{A_1, \ldots, A_d\}$. For each pair $i \neq j$, compute the matrix $[A_i, A_j] = A_iA_j - A_jA_i$ and write it as a linear combination of the A_k. Then the structure constants of $\mathcal{L}$ are given by the $c_{ij}^{(k)}$ defined by $[A_i, A_j] = \sum_k c_{ij}^{(k)} A_k$. Now, the abstract copy $\mathcal{L}'$ of $\mathcal{L}$ can be defined as a d-dimensional vector space with standard basis $v_1 = (1, 0, \ldots, 0), v_2 = (0, 1, 0, \ldots, 0), \ldots, v_d = (0, 0, \ldots, 1)$, and structure constants $c_{ij}^{(k)}$. The abstract isomorphism between $\mathcal{L}$ and $\mathcal{L}'$ is given by $A_i \leftrightarrow v_i$.

C. Structure theory of Lie algebras

Given any two (abstract) Lie algebras $\mathcal{L}_1, \mathcal{L}_2$, their *direct sum* is the Lie algebra $\mathcal{L}_1 \oplus \mathcal{L}_2$ whose underlying vector space is the direct sum of the underlying vector spaces of the $\mathcal{L}_i$. The bracket $[v_1, v_2]$ is that of $\mathcal{L}_i$ if both $v_1, v_2 \in \mathcal{L}_i$, and is zero otherwise, i. e., when $v_1 \in \mathcal{L}_1$ and $v_2 \in \mathcal{L}_2$.

An *ideal* in a Lie algebra is a subspace $I \subseteq \mathcal{L}$ such that $[u, v] \in I$ for any $u \in \mathcal{L}$ and $v \in I$. Ideals are the Lie-algebraic analogue of normal subgroups of groups. Given any ideal, one can form the quotient Lie algebra $\mathcal{L}/I$ whose elements are additive cosets of I, that is, of the form $v + I$; conversely, the kernel of any Lie algebra homomorphism is an ideal.

A Lie algebra is *abelian* if $[u, v] = 0$ for all $u, v \in \mathcal{L}$. Any vector space can thus trivially be given the structure of an abelian Lie algebra. Every subspace of an abelian Lie algebra is an ideal.

0 is the trivial ideal. An ideal in $\mathcal{L}$ is *proper* if it is $\neq \mathcal{L}$. In a direct sum $\mathcal{L} = \mathcal{L}_1 \oplus \mathcal{L}_2$, each $\mathcal{L}_i$ is a proper ideal of $\mathcal{L}$. A Lie algebra is *simple* if it contains no proper non-trivial ideals, and is non-abelian. (This last condition is a convenient, standard convention that excludes the 1-dimensional abelian Lie algebra.) In characteristic zero, a Lie algebra is *semisimple* if and only if it is a direct sum of simple Lie algebras.

Over $\mathbb{C}$, the simple Lie algebras have been completely classified for nearly a century. They fall into four infinite families, referred to as type A_n ($\mathfrak{sl}_n$, consisting of all trace zero $n \times n$ matrices), B_n ($\mathfrak{so}_{2n+1}$, consisting of all $(2n + 1) \times (2n + 1)$ skew-symmetric matrices $M = -M^T$), C_n ($\mathfrak{sp}_{2n}$ consisting of all $2n \times 2n$ matrices M satisfying $JM = -M^TJ$ where $J = \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix}$), and D_n ($\mathfrak{so}_{2n}$), and there are five exceptional simple Lie algebras, known as $\mathfrak{e}_6$, $\mathfrak{e}_7$, $\mathfrak{e}_8$, $\mathfrak{f}_4$, and $\mathfrak{g}_2$. Over any perfect field of characteristic $\neq 2, 3$ the same classification holds for the simple classical Lie algebras (see [12]).

D. Representations

A *representation of a Lie algebra* $\mathcal{L}$ is a homomorphism $\rho\colon \mathcal{L} \to M_n$ for some n. A representation is *faithful* if ρ is injective. Two representations $\rho_1, \rho_2\colon \mathcal{L} \to M_n$ are *equivalent* if there is an invertible $n \times n$ matrix A such that $\rho_1(x) = A\rho_2(x)A^{-1}$ for all $x \in \mathcal{L}$.

Equivalence of representations is similar to, but not the same as, matrix isomorphism of matrix Lie algebras. Given two representations $\rho_1, \rho_2\colon \mathcal{L} \to M_n$, their images $\mathcal{L}_i := \mathrm{Im}(\rho_i)$ are matrix Lie algebras. The representations ρ_i are equivalent if they are conjugate *as maps*, whereas the matrix Lie algebras forget the maps and only care about their images. Lemma IV.2 shows that $\mathcal{L}_1$ and $\mathcal{L}_2$ are matrix isomorphic if and only if ρ_1 and ρ_2 are equivalent up to an automorphism of $\mathcal{L}$, that is, ρ_1 is equivalent to $\rho_2 \circ \alpha$ for some automorphism $\alpha\colon \mathcal{L} \to \mathcal{L}$. These automorphisms are what cause all the computational difficulties, and allow the equivalences with GRAPH ISOMORPHISM and CODE EQUIVALENCE.

A representation $\rho\colon \mathcal{L} \to M_n$ is specified in algorithms by giving the k matrices $\rho(v_i)$ for each basis element v_i of $\mathcal{L}$.

Given two representations $\rho_i \colon \mathcal{L} \to M_{n_i}$ for $i = 1, 2$, their *direct sum* $\rho_1 \oplus \rho_2 \colon \mathcal{L} \to M_{n_1+n_2}$ is defined by the block-matrix:

$$(\rho_1 \oplus \rho_2)(v) = \begin{pmatrix} \rho_1(v) & 0 \\ 0 & \rho_2(v) \end{pmatrix}.$$

The set M_n of $n \times n$ matrices acts on the vector space $\mathbb{F}^n$ by the usual matrix-vector multiplication. Given a subset $S \subseteq M_n$, if $V \subseteq \mathbb{F}^n$ is a subspace such that $S \cdot V \subseteq V$, then V is called an *S-invariant subspace*. The 0 subspace and the whole space $\mathbb{F}^n$ are S-invariant for any S.

A representation $\rho \colon \mathcal{L} \to M_n$ is called *irreducible* if 0 and $\mathbb{F}^n$ are the only $\mathrm{Im}(\rho)$-invariant subspaces. Otherwise a representation is called *reducible*. Note that a reducible representation need not be equivalent to any non-trivial direct sum, as illustrated by the example $\{\begin{pmatrix} 1 & x \\ 0 & 1 \end{pmatrix} : x \in \mathbb{F}\}$. A representation is *completely reducible* if it is (equivalent to) a direct sum of irreducible representations.

If $\mathcal{L} \subseteq M_n$ is a matrix Lie algebra, then it is the image of its own (faithful) representation given by the inclusion (i. e., identity) map $\rho \colon \mathcal{L} \to M_n$. Via this identification, we also apply the terms (ir)reducible to matrix Lie algebras. If $\mathcal{L}$ is a completely reducible matrix Lie algebra, then it is matrix isomorphic to a Lie algebra consisting of block-diagonal matrices, where the restriction to each block is irreducible.

Theorem III.1 (see Theorem III.10 on p. 81 of Jacobson [20])**.** *A matrix Lie algebra $\mathcal{L}$ over $\mathbb{C}$ is completely reducible if and only if $\mathcal{L}$ is isomorphic to the direct sum of a diagonalizable Lie algebra and a semisimple Lie algebra.*

When working over finite fields, our results assume that the matrix Lie algebras are completely reducible; this assumption is necessary, as Jacobson showed that every Lie algebra in positive characteristic has a faithful completely reducible representation and a faithful representation that is not completely reducible [21] (see also [20, Chapter VI]).

We can also make the above theorem effective. The following result is surely known, but we could not find an explicit reference to it. The proof can be found in the full version of this paper [9, Chapter 4], and is based on the proof of Theorem III.1 in Jacobson [20]; the key algebraic tool is Whitehead's Lemma, which yields an efficient algorithm for finding the complement of a sub-representation.

Theorem III.2. *Given a completely reducible $\mathcal{L} \subseteq M_n$, one can find in $poly(n)$ f-time a matrix A so that $A\mathcal{L}A^{-1}$ is the direct sum of an abelian diagonal Lie algebra and a semisimple Lie algebra, where the semisimple part consists of block-diagonal matrices, each block irreducible.*

E. Inner and Outer Automorphisms

The collection of automorphisms of a Lie algebra $\mathcal{L}$ form a group $\mathrm{Aut}(\mathcal{L})$ under composition of maps. Given a Lie algebra $\mathcal{L}$ and $x \in \mathcal{L}$, the Jacobi identity implies that the map $\mathrm{ad}_x \colon \mathcal{L} \to \mathcal{L}$ defined by $\mathrm{ad}_x(y) := [x, y]$ is a homomorphism. If $\mathrm{ad}_x^k := \mathrm{ad}_x \circ \cdots \circ \mathrm{ad}_x$ is the zero map for k sufficiently large, then $\exp(\mathrm{ad}_x) := I + \mathrm{ad}_x + \frac{1}{2}\mathrm{ad}_x^2 + \cdots + \frac{1}{(k-1)!}\mathrm{ad}_x^{k-1}$ is an automorphism of $\mathcal{L}$. Automorphisms arising in this way are called *inner*. The inner automorphisms form a normal subgroup $\mathrm{Inn}(\mathcal{L}) \leq \mathrm{Aut}(\mathcal{L})$. The quotient group $\mathrm{Aut}(\mathcal{L})/\mathrm{Inn}(\mathcal{L})$ is called the *outer automorphism* group, denoted $\mathrm{Out}(\mathcal{L})$.

The outer automorphism groups of the simple Lie algebras are completely known: $\mathrm{Out}(\mathfrak{sl}_n) = S_2$, $\mathrm{Out}(\mathfrak{so}_{2n}) = S_2$ when $n \neq 4$, $\mathrm{Out}(\mathfrak{so}_8) = S_3$ $\mathrm{Out}(\mathfrak{e}_6) = S_2$, and all the others are trivial. We mention that unique nontrivial outer automorphism of $\mathfrak{sl}_n$ happens to send representations to equivalent representations (see the next section).

F. Twisting representations by automorphisms

Given an automorphism $\alpha \in \mathrm{Aut}(\mathcal{L})$ and a representation $\rho \colon \mathcal{L} \to M_n$, we get another representation $\rho \circ \alpha \colon \mathcal{L} \to M_n$, given by $(\rho \circ \alpha)(v) = \rho(\alpha(v))$. We call $\rho \circ \alpha$ the *twist* of the representation ρ by the automorphism α.

The twist $\rho \circ \alpha$ and ρ need not be equivalent as representations, despite having the same image (since α is an automorphism, it is onto, so $\mathrm{Im}(\rho \circ \alpha) = \mathrm{Im}(\rho)$). However, for semisimple Lie algebras, twisting by inner automorphisms does yield equivalent representations:

Lemma III.3 (Lemma 8.5.1 in de Graaf [11])**.** *Let $\rho \colon \mathcal{L} \to M_n$ be a representation of a semisimple Lie algebra $\mathcal{L}$ and let $\alpha \in \mathrm{Inn}(\mathcal{L})$. Then $\rho \circ \alpha$ is equivalent to ρ.*

de Graaf's proof does not work in positive characteristic; in the full version [9, Chapter 4] we show how to modify the proof to recover the result in sufficiently large characteristic.

Since twisting a representation by an inner automorphism sends it to an equivalent representation, the outer automorphism group $\mathrm{Out}(\mathcal{L})$ acts on the set of representations-up-to-equivalence. If $\alpha \in \mathrm{Out}(\mathcal{L})$, we denote the image of ρ under the action of α by ρ^α. Equivalently, let $\alpha_* \in \mathrm{Aut}(\mathcal{L})$ be a representative of $\alpha \in \mathrm{Out}(\mathcal{L})$; then ρ^α is the equivalence class of $\rho \circ \alpha_*$, and by the lemma, this equivalence class is independent of the choice of representative α_*.

IV. SEMISIMPLE MATISOLIE AND GRAPH ISOMORPHISM

Theorem IV.1. GRAPHISO $\leq_m$ SEMISIMPLE MATISOLIE $\leq_m^f$ GRAPHISO. *In particular, the two are f-equivalent.*

Proof: We break the proof into four lemmas. By Lemma IV.2, SEMISIMPLE MATISOLIE is f-equivalent to deciding whether two representations of a semisimple Lie algebra are equivalent up to outer automorphism. By Lemma IV.3, the latter problem reduces to a special case of TWISTED CODE EQUIVALENCE WITH MULTIPLICITIES, which we refer to as PROBLEM A. Finally, Lemma IV.4 reduces PROBLEM A to GRAPH ISOMORPHISM, and Lemma IV.5 reduces GRAPH ISOMORPHISM to SEMISIMPLE MATISOLIE. ∎

Lemma IV.2 (de Graaf[2])**.** SEMISIMPLE MATISOLIE *is f-equivalent to—nearly just a restatement of—the following problem:*

> ***Problem:*** OUTER EQUIVALENCE OF LIE ALGEBRA REPRESENTATIONS
> ***Input:*** *Two faithful representations $\rho_1, \rho_2 \colon \mathcal{L} \to M_n$ of a semisimple (abstract) Lie algebra $\mathcal{L}$..*
> ***Output:*** *An outer automorphism $\alpha \in \mathrm{Out}(\mathcal{L})$ such that ρ_1^α is equivalent to ρ_2, or "the two representations are not equivalent up to automorphism."*

Proof: Suppose $\mathcal{L}_1, \mathcal{L}_2 \subseteq M_n$ is are two semisimple matrix Lie algebras. Using techniques given in de Graaf [11, §5.11], we can determine if the $\mathcal{L}_i$ are isomorphic as abstract Lie algebras; if not, they are not matrix isomorphic, or if so, we can construct an abstract Lie algebra $\mathcal{L}$ that they are isomorphic to together with isomorphisms $\rho_i \colon \mathcal{L} \to \mathcal{L}_i$ for $i = 1, 2$ (see Section III-B). Since $\mathcal{L}_i \subseteq M_n$, the ρ_i are faithful representations of $\mathcal{L}$. We claim that the ρ_i are equivalent up to $\mathrm{Out}(\mathcal{L})$ if and only if the $\mathcal{L}_i$ are matrix isomorphic.

Suppose $\mathcal{L}_2 = A\mathcal{L}_1A^{-1}$. Let $c_A \colon M_n \to M_n$ be defined by $c_A(X) = AXA^{-1}$. Then $\alpha = \rho_2^{-1} \circ c_A \circ \rho_1$ is a map from $\mathcal{L}$ to $\mathcal{L}$. Since the ρ_i are isomorphisms, and $c_A|_{\mathcal{L}_1} \colon \mathcal{L}_1 \to \mathcal{L}_2$ is an isomorphism, the composition α is an automorphism of $\mathcal{L}$. Then $\rho_2 \circ \alpha = \rho_2 \circ \rho_2^{-1} \circ c_A \circ \rho_1 = c_A \circ \rho_1$, which is by definition equivalent to ρ_1. By the discussion following Lemma III.3, $\rho_2^{\overline{\alpha}}$ is thus equivalent to ρ_1, where $\overline{\alpha}$ is the outer automorphism corresponding to α.

Conversely, suppose $\rho_1^{\overline{\alpha}}$ is equivalent to ρ_2 for some outer automorphism $\overline{\alpha}$. Let $\alpha \in \mathrm{Aut}(\mathcal{L})$ be a representative of $\overline{\alpha}$; then there is an invertible matrix A such that $\rho_2 = c_A \circ \rho_1 \circ \alpha$. Then we have

$$\mathcal{L}_2 = \mathrm{Im}(\rho_2) = \mathrm{Im}(c_A \circ \rho_1 \circ \alpha) = c_A(\mathrm{Im}(\rho_1 \circ \alpha)).$$

Since α is an automorphism it is onto, so $\mathrm{Im}(\rho_1 \circ \alpha) = \mathrm{Im}(\rho_1) = \mathcal{L}_1$, and we have $\mathcal{L}_2 = A\mathcal{L}_1A^{-1}$.

The preceding argument gives a reduction from SEMISIMPLE MATISOLIE to OUTER EQUIVALENCE OF LIE ALGEBRA REPRESENTATIONS. The reduction in the other direction is as follows: suppose $\rho_1, \rho_2 \colon \mathcal{L} \to M_n$ are two faithful representations of a semisimple Lie algebra $\mathcal{L}$. We reduce this to the instance of SEMISIMPLE MATISOLIE given by $\mathcal{L}_i = \mathrm{Im}(\rho_i)$ $(i = 1, 2)$. The proof that this is a reduction is identical to the proof above. ∎

Lemma IV.3. OUTER EQUIVALENCE OF LIE REPRESENTATIONS *f-Karp-reduces to the following problem:*

> ***Problem:*** PROBLEM A
> ***Input:*** *Two $r \times s$ integer matrices M_1, M_2; a partition of the columns into consecutive ranges $[1, \ldots, k_1], [k_1 + 1, \ldots, k_1 + k_2], \ldots [k_1 + \cdots + k_{t-1} + 1, \ldots, s]$; for each range, a group G_ℓ acting on the integers appearing in the corresponding columns, where each G_ℓ is abstractly isomorphic to one of: 1, S_2, or S_3.*
> ***Output:*** *A permutation $\pi \in S_r$, a permutation $\sigma \in S_{k_1} \times S_{k_2} \times \cdots \times S_{k_t}$, and for each column an element g_j in the group G_ℓ associated to that column range, such that for all i, j, $M_1(i, j) = g_j(M_2(\pi(i), \sigma(j)))$, or "the matrices are not equivalent." In other words, after applying π to the rows, σ to the columns, and each g_j to the values of the entries in the j-th column, M_1 and M_2 become equal.*

Proof: Let $\rho_1, \rho_2 \colon \mathcal{L} \to M_n$ be two faithful representations of an (abstract) semisimple Lie algebra $\mathcal{L}$. Compute the direct sum decomposition of $\mathcal{L}$; suppose it is $\mathcal{L} = \mathcal{L}_{1,1} \oplus \cdots \oplus \mathcal{L}_{1,k_1} \oplus \mathcal{L}_{2,1} \oplus \cdots \oplus \mathcal{L}_{2,k_2} \oplus \cdots \oplus \mathcal{L}_{t,k_t}$ where each $\mathcal{L}_{a,b}$ is a simple summand of $\mathcal{L}$, and the $\mathcal{L}_{a,b}$ are grouped by isomorphism type, so that $\mathcal{L}_{a_1,b_1}$ and $\mathcal{L}_{a_2,b_2}$ are isomorphic if and only if $a_1 = a_2$. For each a, let $\mathcal{L}_a$ be a simple Lie algebra isomorphic to $\mathcal{L}_{a,b}$ for all b.

Note: the above step uses algorithms from de Graaf [11] that use root-finding. Over finite fields, which we only consider in the full version, it also needs Ryba's algorithm for finding a split Cartan subalgebra [13].

To each ρ_i we will associate a matrix M_i, as well as the other data necessary for PROBLEM A. The columns correspond to the direct summands $\mathcal{L}_{a,b}$, and the column partition is along the isomorphism types of the summands.

Next, we define the permutation groups G_ℓ. To each simple type $\mathcal{L}_\ell$, we fix once and for all an encoding of its representations as integers; both the encoding and decoding should be polynomial-time. That this can be done follows from the standard description of the representations of the simple Lie algebras. The integer 0 will always stand for the (trivial) zero representation. The permutation action of $\mathrm{Out}(\mathcal{L}_\ell)$ on the representations of $\mathcal{L}_\ell$, encoded as integers, can be easily computed, as follows. Given $\overline{\alpha} \in \mathrm{Out}(\mathcal{L}_\ell)$ and an integer, convert it to the corresponding representation as above. This representation is a linear map $\mathcal{L}_\ell \to M_n$ for some n. Pre-compose this map with a representative $\alpha \in \mathrm{Aut}(\mathcal{L}_\ell)$ of $\overline{\alpha}$; this can be done because the outer automorphisms of all simple Lie algebras are known explicitly and are easy to compute. For example, the unique outer automorphisms of $\mathfrak{sl}_n$, the trace zero matrices, is given by the map $A \mapsto -A^T$. The outer automorphism groups of simple Lie algebras are all trivial, S_2, or S_3. Finally, convert this new, twisted-by-α representation back to an integer. The group G_ℓ associated to the ℓ-th isomorphism type (=ℓ-th column grouping) is then $\mathrm{Out}(\mathcal{L}_\ell)$, and the action on the

[2]This lemma is essentially present in de Graaf [11]; see the discussion at the end of his Section 8.5. However, de Graaf's discussion is presented in terms of weights and the choice of Cartan subalgebra, whereas the aspect we wish to highlight requires no mention of these topics.

integers is the action just described.

Finally, we describe the rows and the entries of the matrices M_i. Decompose the representations ρ_i into their direct sum decompositions $\rho_i = \rho_{i,1} \oplus \cdots \oplus \rho_{i,r}$, where each $\rho_{i,r}$ is an irreducible representation of $\mathcal{L}$ (see Corollary III.2). The q-th row of M_i corresponds to the irreducible representations $\rho_{i,q}$. An irreducible representation of a direct sum of Lie algebras is completely specified by its restriction to each summand. Hence, the representation $\rho_{i,q}$ is specified by a representation of each summand $\mathcal{L}_{a,b}$, that is, an integer in each column.

Since the outer automorphism group of $\mathcal{L}$ is $\prod_{a=1}^{t} \mathrm{Out}(\mathcal{L}_a) \wr S_{k_a} = \left(\prod_{a=1}^{t} \mathrm{Out}(\mathcal{L}_a)^{k_a}\right) \rtimes \left(\prod_{a=1}^{t} S_{k_a}\right)$, the representations ρ_1, ρ_2 are equivalent up to an outer automorphism if and only if there is a permutation of the columns (=direct summands of the Lie algebra), for each column an element $g_\ell \in G_\ell$ (=an outer automorphism of each direct summand), and a permutation of the rows (=irreducible constituents of ρ_i) that will make M_1 equal to M_2. Conversely, any such equivalence of M_1 and M_2 according to PROBLEM A corresponds to an outer automorphism of $\mathcal{L}$ that makes ρ_1 and ρ_2 equivalent. ■

Lemma IV.4. PROBLEM A *reduces to* GRAPH ISOMORPHISM.

Proof sketch: This is a fun exercise we invite the reader to try for him- or herself. The full proof is included in the full version of the paper, and goes by encoding the actions of the column groups G_ℓ via graph gadgets. ■

The encoding mentioned in the above proof sketch is done specifically for the groups 1, S_2, and S_3. We suspect that the same idea can be extended to show that TWISTED CODE EQUIVALENCE in general, as defined in Codenotti [22], Karp-reduces to GRAPH ISOMORPHISM.

Lemma IV.5. GRAPHISO $\leq_m$ SEMISIMPLE MATISOLIE.

Proof: Let G_1, G_2 be two graphs with no isolated vertices, and let D_i be the 0-1 incidence matrix of G_i, where the rows correspond to edges and the columns correspond to vertices. The G_i are isomorphic if and only if there is a permutation of the rows and the columns that makes the D_i equal. Then (D_1, D_2) is an instance of PROBLEM A where the column partition is trivial, and the column groups G_ℓ are also trivial. We show how to reduce such an instance of PROBLEM A to OUTER EQUIVALENCE OF LIE ALGEBRA REPRESENTATIONS, and hence to SEMISIMPLE MATISOLIE. Note that the reduction from OUTER EQUIVALENCE to MATISOLIE is deterministic, as it simply takes each representation to its image.

Given an instance of PROBLEM A as above—in particular, it only contains the entries 0 and 1, it contains exactly two non-zero entries per row, every column contains a non-zero entry (=no isolated vertices in the graphs), and the column partition and column groups are all trivial—let $\mathcal{L} = \mathfrak{sl}_2^{\oplus n}$, where n is the number of vertices of the G_i (=columns of the matrices) and $\mathfrak{sl}_2$ is the simple Lie algebra of 2×2 trace zero matrices. Let a 1 in the matrix D_i correspond to the adjoint representation of $\mathfrak{sl}_2$, which is faithful and has dimension 3. Then, by reversing the reduction in Lemma IV.3, we get an instance of OUTER EQUIVALENCE OF LIE ALGEBRA REPRESENTATIONS.

Since each column contains a non-zero entry, these representations are faithful. Since each row contains exacty two 1's, the corresponding irreducible representation has dimension $3^2 = 9$, hence the representations we get are matrices of dimension $9m \times 9m$, where m is the number of edges of G_i. Since $\mathfrak{sl}_2^{\oplus n}$ is generated by $3n$ elements, the representations can be specified by $3n \times (9m)^2$ numbers, which is polynomial in the size of the original graphs.

Finally, $\mathrm{Out}(\mathfrak{sl}_2)$ acts trivially on the representations of $\mathfrak{sl}_2$ up to equivalence, so the corresponding column groups are trivial, as desired. ■

Theorem IV.6. SEMISIMPLE MATISOLIE *of $n \times n$ matrices can be solved in polynomial f-time, when the Lie algebras consist of $O(\log n)$ simple direct summands.*

Proof: If there are only $O(\log n / \log\log n)$ simple summands, then an elementary brute-force approach to PROBLEM A works in $poly(n)$ time, since the number of outer automorphisms is $poly(n)$. However, when there are $O(\log n)$ simple summands, the number of outer automorphisms can be $n^{O(\log n)}$, so we instead use a more sophisticated approach to TWISTED CODE EQUIVALENCE, due to Babai, Codenotti, and Qiao [23] (cf. [22, Theorem 4.2.1]). PROBLEM A is in fact a special case of TWISTED CODE EQUIVALENCE WITH MULTIPLICITIES, in which each row corresponds to a codeword. On the instance of PROBLEM A corresponding to semisimple Lie algebras with $O(\log n)$ simple summands, their algorithm runs in $poly(n)$ time. Translating between their terminology and ours, the size of the code is the number of rows of the M_i, which is the number irreducible representations of the $\mathcal{L}_i$, which is at most n, the size of the original matrices. Furthermore, the column groups G_i all have bounded size. These two facts together imply that their algorithm runs in $poly(n)$ time. ■

Theorem IV.7. SEMISIMPLE MATISOLIE *of $n \times n$ matrices can be solved in polynomial f-time, when the Lie algebras consist of $O(\log n / \log\log n)$ irreducible representations, and unboundedely many simple direct summands, at most $O(\log(n))$ of which have nontrivial outer automorphism actions on their representations.*

Three of the four infinite families of simple Lie algebras have trivial outer automorphism action on their representations, as well as four of the five exceptional simple Lie algebras (see Section III-E).

Proof: In this case, the M_i in the instance of PROBLEM A have only $f(n) \leq O(\log n / \log\log n)$ rows. Although the

size of the automorphism group may be super-polynomial, there are only polynomially many row permutations, so we only have to handle exhaustively the outer automorphisms in each column, and not the permutations between the columns. Specifically, try each combination outer automorphism of each column; since there are only $O(\log n)$ columns with nontrivial outer automorphisms, and the outer automorphism group of a simple Lie algebra has size at most 6, there are only $poly(n)$ possibilities. For each such possibility, try each of the $poly(n)$ many permutations of the rows, and for each check whether the set of columns of M_1 is equal to the set of columns of M_2. ∎

V. Abelian plus semisimple MatIsoLie

In this section, we describe how the algorithms and reductions for the abelian diagonalizable and semisimple cases fit into a single general framework and can be combined to handle the case of a direct sum of an abelian diagonalizable matrix Lie algebra with a semisimple matrix Lie algebra. In characteristic zero, this class of matrix Lie algebras is exactly the class of completely reducible matrix Lie algebras (cf. Theorem III.1).

Lemma V.1. *Lemma IV.2 applies to the class of abelian Lie algebras and to the class of completely reducible matrix Lie algebras.*

Proof: The proof of Lemma IV.2 only required two ingredients: that the isomorphism problem for abstract Lie algebras of the class under consideration be efficiently solvable, and that twisting a representation by an inner automorphism leads to an equivalent representation. Both of these ingredients hold for abelian Lie algebras: two abelian Lie algebras are abstractly isomorphic if and only if they have the same dimension, and abelian Lie algebras have no non-trivial inner automorphisms.

Similarly, a completely reducible matrix Lie algebra $\mathcal{L}$ is a direct sum $\mathcal{A} \oplus \mathcal{S}$ where $\mathcal{A}$ is abelian diagonalizable and $\mathcal{S}$ is semisimple. The isomorphism problem for this class of Lie algebras is solvable in polynomial f-time. Finally, $\mathrm{Inn}(\mathcal{L}) = \mathrm{Inn}(\mathcal{A}) \times \mathrm{Inn}(\mathcal{S}) \cong \mathrm{Inn}(\mathcal{S})$ since abelian Lie algebras have no non-trivial inner automorphisms. Hence twisting a representation by an inner automorphism leads to an equivalent representation. ∎

Although it was not originally phrased this way, we can now see that the algorithms and equivalences for DIAGONALIZABLE MatIsoLie follow the same lines as those for SEMISIMPLE MatIsoLie. The main difference is that the outer automorphism group of a d-dimensional abelian Lie algebra is the full general linear group GL_d of $d \times d$ invertible matrices—leading to LINEAR CODE EQUIVALENCE—whereas the outer automorphism group of a semisimple Lie algebra is close to S_n—leading to GRAPH ISOMORPHISM.

Furthermore, we can view Babai's reduction (see [19, Theorem 7.1]) from CODE EQUIVALENCE as a sort of "list normal form" algorithm for the action of GL_d by automorphisms. Since GL_d acts by change of basis, we would like reduced row echelon form (rref) to be a normal form for this action. However, since one may permute the coordinates in CODE EQUIVALENCE, computing rref requires first picking the pivots. Babai's algorithm picks these pivots in all $\binom{n}{d}$ possible ways, reduces to rref, and then uses GRAPH ISOMORPHISM to handle the permutation action on the remaining coordinates of the code. Combining these techniques yields:

Theorem V.2. COMPLETELY REDUCIBLE MATISOLIE, *with an abelian diagonalizable part of dimension a, s simple direct summands, and r irreducible representation constituents f-reduces to $\binom{r}{a}$ instances of* PROBLEM A *of size $r \times s$. In particular,* COMPLETELY REDUCIBLE MATISOLIE *of $n \times n$ matrices can be solved in $poly(n)$ f-time under either of the following conditions:*

- *$a = O(\log n)$, $r = O(1)$, s unbounded, and the number of simple summands with non-trivial outer automorphism action is at most $O(\log n)$;*
- *$a = O(1)$, r unbounded, $s = O(\log n)$.*

VI. Application to equivalence of polynomials

Corollary VI.1. *Given the Lie algebra of the symmetry group of a polynomial f on n^2 variables, one can determine whether f is linearly equivalent to $\det_n$ in* deterministic *$poly(n)$ time with a root-finding oracle.*

As in Kayal [8], this yields an algorithm for testing LINEAR EQUIVALENCE to the determinant. Kayal gave randomized algorithms for this problem; the algorithm for LINEAR EQUIVALENCE based on Corollary VI.1 only needs randomness to compute the Lie algebra in the first place, but thereafter is deterministic, unlike Kayal's algorithm. We now show that this is essentially necessary, unless strong lower bounds can be proven.

Remark VI.2 (On the necessity of POLYNOMIAL IDENTITY TESTING)**.** We will show that computing the Lie algebra of the symmetry group of a polynomial is (Karp-)*equivalent* to POLYNOMIAL IDENTITY TESTING (PIT), and hence cannot be derandomized without proving significant lower bounds [24]. Kayal [8, Lemma 26] shows how to compute the Lie algebra of the symmetry group of a polynomial given as a black-box, using the algorithm from [25] for computing the linear dependincies between a set of polynomials. Kayal [25] noted that computing such linear dependencies reduces to the search version of PIT. The search and decision versions of PIT are equivalent for low-degree functions. Conversely, a polynomial is constant if and only if its symmetry group consists of all invertible transformations of the variables if and only if the Lie algebra of its symmetry group consists of all matrices. Once this has been determined, evaluating the polynomial at any single point will determine whether it is

zero or a non-zero constant. Hence, PIT stands in the way of derandomizing any approach to AFFINE EQUIVALENCE based on the Lie algebras of the symmetry groups of polynomials.

Remark VI.3 (On the necessity of other randomness)**.** For LINEAR EQUIVALENCE to the determinant, we also use a root-finding oracle. Over $\mathbb{C}$ this introduces no additional randomness, so by the previous remark over $\mathbb{C}$ we have essentially derandomized Kayal's algorithm as much as is currently possible.

Over finite fields, we currently use additional randomness—in constructing a split Cartan subalgebra [13] (see the full paper for details)—so although we avoid some randomness Kayal uses, there is other randomness that might be removed without derandomizing PIT. We nonetheless recover and generalize Kayal's result over finite fields.

In some sense, we have thus derandomized Kayal's algorithm as far as is possible in the black-box setting without derandomizing PIT. Kayal uses randomization at several points, not just in the computation of the Lie algebra of the symmetry group, and we derandomize those using our deterministic f-algorithm for SEMISIMPLE MATISOLIE.

In the dense, non-black-box setting this also represents an improvement: from $2^{O(n^2)}$ to $2^{O(n \log n)}$. This is essentially optimal, since a generic function that is equivalent to $\det_n$ will include nearly all monomials of degree n in n^2 variables, of which there are $2^{\Theta(n \log n)}$. By the dense setting we mean that f is given as a list of coefficients of all monomials of degree n in n^2 variables. Naive derandomization of Kayal's algorithm takes time $2^{O(n^2)}$, since step (ii) of his Section 6.2.1 guesses a random element of a space of dimension $\Theta(n^2)$. Similarly, testing affine equivalence to the determinant can be solved using quantifier elimination (see, e. g., Basu, Pollack, and Roy [26, Ch. 14]), again in time $2^{O(n^2)}$, because the witness to equivalence is an $n \times n$ matrix together with an n-dimensional vector. However, computing the Lie algebra of the symmetry group of f only requires solving a linear system of n^2 equations in a number of variables equal to the number of monomials possible, which is roughly $\binom{n^2}{n} \leq n^{2n} = 2^{O(n \log n)}$.

Proof of Corollary VI.1: The Lie algebra of the symmetry group of $\det_n$ is $\mathbb{C} \oplus \mathfrak{sl}_n \oplus \mathfrak{sl}_n$, which has only two simple factors. By Theorem V.2 we can test if the Lie algebra of the symmetry group of f is matrix isomorphic to that of $\det_n$. If it is, then act on f by the conjugating matrix so that the Lie algebra is now equal to that of $\det_n$. Any function whose symmetry group has a Lie algebra equal (as sets of matrices) to that of the determinant is a scalar multiple of the determinant (see, e. g., [9, Chapter 3]). Note that we have combined here all three main steps of Kayal's algorithm into a single reduction to MATISOLIE: Kayal uses the Lie algebra to reduce to permutational and scaling equivalence, then solves permutational equivalence and scaling equivalence separately. ■

VII. CONCLUSION AND FUTURE WORK

MATRIX ISOMORPHISM OF MATRIX LIE ALGEBRAS arises in Geometric Complexity Theory and the affine equivalence problem. We solved MATISOLIE over $\mathbb{C}$ in polynomial f-time for several important classes of Lie algebras—namely diagonalizable, semisimple, and completely reducible (=diagonalizable $\oplus$ semisimple)—under various quantitative constraints. We showed that without these quantitative constraints, these cases of MATISOLIE all become at least as hard as GRAPH ISOMORPHISM.

The completely reducible case is not far from the general case, though significant obstacles remain. Levi's Theorem says that every Lie algebra in characteristic zero is the semidirect product of a solvable Lie algebra by a semisimple one; a solvable Lie algebra is an iterated extension of abelian Lie algebras. The completely reducible case, which we resolved, restricts the solvable part to be abelian, and restricts the semidirect product to be direct. The complexity of MATISOLIE in general remains open, but we believe the following is an achievable next target:

Open Question VII.1. What is the complexity of MATISOLIE for matrix Lie algebras that are *semi*direct products of abelian by semisimple?

We essentially derandomized Kayal's algorithm for testing equivalence to the determinant, except for a part of the algorithm that is equivalent to POLYNOMIAL IDENTITY TESTING and root-finding. It would be nice to know whether there is a way around these, though we suspect there is not:

Open Question VII.2. Show that testing equivalence to the determinant is as hard as PIT and root-finding, or give a deterministic polynomial-time algorithm for it in the black-box setting. In the dense setting, can equivalence to the determinant be tested in time $poly(t)$ where t is the number of non-zero monomials of the input function?

Finally, there are two avenues for futher progress on AFFINE EQUIVALENCE using MATISOLIE. First, although GRAPHISO-hardness may seem to be the "final" word in the short term, in the application to AFFINE EQUIVALENCE we may be able to avoid GRAPHISO altogether. MATISOLIE is most directly useful for testing affine equivalence to symmetry-characterized functions such as the determinant. A function f is *symmetry-characterized* if for any function g, if g has the same symmetries as f—that is, $f(A\mathbf{x}) = f(\mathbf{x})$ implies $g(A\mathbf{x}) = g(\mathbf{x})$—then g is a scalar multiple of f. Not every Lie algebra can arise as the Lie algebra of the symmetries of a symmetry-characterized function. It is possible that the properties of such Lie algebras are strong enough to avoid GRAPHISO. Second, in addition to the Lie algebra of the symmetries of a function, a function may

have a finite group of symmetries "sitting on top of" the Lie algebra.

Open Question VII.3. What is the complexity of testing matrix isomorphism of finite groups of symmetries, arising from symmetry-characterized functions?

ACKNOWLEDGMENT

The author thanks the following people for useful discussions: N. Kayal, P. Koiran, A. Chattopadhyay, J. M. Landsberg, S. Kumar, and J. Weyman. Many of these conversations took place at the Brown-ICERM Workshop on Mathematical Aspects of P vs. NP and its Variants in August 2011, for which the author thanks ICERM and the organizers of the workshop—J. M. Landsberg, S. Basu, and J. M. Rojas—for the invitation and support to attend the workshop. The author thanks L. Fortnow, K. Mulmuley and B. Farb for their discussions, support, and advice. The author finds it incredibly useful and enjoyable to talk through mathematics with others, and it is his great pleasure to thank B. Farb, T. Church, I. Shipman, and J. Blasiak for not only useful and interesting discussions of this work, but also for their infectious enthusiasm for and injection of fruitful new ideas into this work. In particular, Blasiak helped the author clarify his thoughts and together realize the equivalence with GRAPH ISOMORPHISM. Finally, this work was partially supported by K. Mulmuley's NSF Grant CCF-1017760, L. Fortnow *et al.*'s NSF Grant DMS-0652521 and fellowships from the U. Chicago Department of Computer Science.

REFERENCES

[1] P. J. Olver, *Applications of Lie groups to differential equations*, 2nd ed. Springer-Verlag, 1993.

[2] W.-H. Steeb, *Continuous symmetries, Lie algebras, differential equations and computer algebra*, 2nd ed. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2007.

[3] H. Georgi, *Lie algebras in particle physics*, ser. Frontiers in Physics. Benjamin/Cummings Publishing Co., 1982, vol. 54.

[4] W. Fulton and J. Harris, *Representation theory: a first course*. Springer-Verlag, 1991.

[5] C. Chevalley, *Theory of Lie Groups. I.* Princeton University Press, 1946.

[6] A. L. Onishchik and È. B. Vinberg, *Lie groups and algebraic groups*, ser. Springer Series in Soviet Mathematics. Springer-Verlag, 1990.

[7] K. D. Mulmuley and M. Sohoni, "Geometric complexity theory I: an approach to the P vs. NP and related problems," *SIAM J. Comput.*, vol. 31, no. 2, pp. 496–526, 2001.

[8] N. Kayal, "Affine projections of polynomials," Electronic Colloq. Comput. Complexity, Tech. Rep. TR11-061, 2011.

[9] J. A. Grochow, "Symmetry and equivalence relations in classical and geometric complexity theory," Ph.D. dissertation, The University of Chicago, Chicago, IL, expected June 2012.

[10] E. Petrank and R. M. Roth, "Is code equivalence easy to decide?" *IEEE Trans. Inf. Theory*, vol. 43, pp. 1602–1604, 1997.

[11] W. A. de Graaf, *Lie algebras: theory and algorithms*. Amsterdam: North-Holland Publishing Co., 2000.

[12] G. B. Seligman, *Modular Lie algebras*. Springer-Verlag, 1967.

[13] A. J. E. Ryba, "Computer construction of split Cartan subalgebras," *J. Algebra*, vol. 309, no. 2, pp. 455–483, 2007.

[14] L. Blum, M. Shub, and S. Smale, "On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines," *Bull. Amer. Math. Soc. (N.S.)*, vol. 21, no. 1, pp. 1–46, 1989.

[15] E. R. Berlekamp, "Factoring polynomials over finite fields," *Bell System Tech. J.*, vol. 46, pp. 1853–1859, 1967.

[16] ——, "Factoring polynomials over large finite fields," *Math. Comp.*, vol. 24, pp. 713–735, 1970.

[17] C. A. Neff and J. H. Reif, "An efficient algorithm for the complex roots problem," *J. Complexity*, vol. 12, no. 2, pp. 81–115, 1996.

[18] E. Kaltofen, "Polynomial factorization 1987–1991," in *LATIN '92*, ser. Lecture Notes in Computer Science, I. Simon, Ed. Springer Berlin / Heidelberg, 1992, vol. 583, pp. 294–313.

[19] L. Babai, P. Codenotti, J. A. Grochow, and Y. Qiao, "Code equivalence and group isomorphism," in *ACM-SIAM Symposium on Discrete Algorithmas (SODA11)*, 2011.

[20] N. Jacobson, *Lie algebras*. Interscience Publishers (a division of John Wiley & Sons), 1962.

[21] ——, "A note on Lie algebras of characteristic p," *Amer. J. Math.*, vol. 74, pp. 357–359, 1952.

[22] P. Codenotti, "Testing isomorphism of combinatorial and algebraic structures," Ph.D. dissertation, University of Chicago, Chicago, IL, 2011.

[23] L. Babai, P. Codenotti, and Y. Qiao, "Testing isomorphism of groups with no abelian normal subgroups," 2011, in preparation.

[24] V. Kabanets and R. Impagliazzo, "Derandomizing polynomial identity tests means proving circuit lower bounds," *Comput. Complexity*, vol. 13, no. 1-2, pp. 1–46, 2004.

[25] N. Kayal, "Efficient algorithms for some special cases of the polynomial equivalence problem," in *ACM-SIAM Symposium on Discrete Algorithmas (SODA11)*, 2011.

[26] S. Basu, R. Pollack, and M.-F. Roy, *Algorithms in real algebraic geometry*, 2nd ed. Berlin: Springer-Verlag, 2006.