On Learning and Lower Bound Problems Related to the Iterated Matrix Multiplication Polynomial

A THESIS
SUBMITTED FOR THE DEGREE OF
Doctor of Philosophy
IN THE
Faculty of Engineering

BY
Vineet Nair

Computer Science and Automation
Indian Institute of Science
Bangalore – 560 012 (INDIA)

November, 2019
Declaration of Originality

I, Vineet Nair, with SR No. 04-04-04-14-12-15-1-13189 hereby declare that the material presented in the thesis titled

On Learning and Lower Bound Problems Related to the Iterated Matrix Multiplication Polynomial

represents original work carried out by me in the Department of Computer Science and Automation at Indian Institute of Science during the years 2015–2019.

With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Date: Student Signature

In my capacity as supervisor of the above-mentioned work, I certify that the above statements are true to the best of my knowledge, and I have carried out due diligence to ensure the originality of the report.

Advisor Name: Chandan Saha Advisor Signature
Acknowledgements

I have spent a little more than 6 years in IISc, including two years of Masters, and my time at IISc has shaped my personality the most. I have learnt important life lessons in these years and I am grateful to many people who have contributed to this journey. This acknowledgement is a small thank you to their efforts that have helped me grow as a person.

I am indebted to my advisor Chandan Saha who has advised and guided me throughout, especially on occasions when my PhD journey was becoming tough. His tough but constructive criticism has helped me improve my skill-set in research. I have had several discussions with him on academics, personal and philosophical issues, and his views have helped me garner a different perspective on them. I am thankful to Chandan for this and for everything else that he has done for me during my PhD and Masters years.

During my two years of Masters, and the initial years of PhD I found the courses from math department difficult to cope with. This changed when I started interacting with Dilip Patil. In my experience teachers like him are rare and I am lucky to have him as my teacher. His courses have taught me how to study, how to write, and most importantly how to think when I am stuck on a problem. I have tried to use his methods while studying other subjects and they have worked wonders for me. I am thankful to Dilip for all these lessons and devoting his time in improving my confidence over mathematical subjects.

I am thankful to my collaborators Neeraj Kayal, Chandan Saha and Sébastien Tavenas. I have learnt a lot by interacting with them and also seeing them interact. I thank Neeraj and Chandan for always encouraging and motivating me to do good research. I am thankful to Rohit Gurjar and Gaurav Sinha with whom I have had several discussions on academic and other problems.

I am thankful to all the professors at the Computer Science and Automation department
Acknowledgements

whose courses I have attended and learned a lot from. I am thankful to Siddharth Barman and Anand Louis, who were in my year end review and comprehensive panel, for asking interesting questions which helped improve my understanding. I have learnt a lot through the engaging discussions I have had with my lab-mates: Sumant Hegde, Abhijat Sharma, Nikhil Gupta and Janaky Murthy. Special thanks to Nikhil for being a close confidante and for many interesting discussions on research and philosophy. I have also been lucky to learn and discuss problems in online learning with Vishakha Patil and Ganesh Ghalme in the past year.

The lonely and monotonous life in IISc is made enjoyable by the friends you make here. I have been lucky to have made a good friend in Mayank Tiwari. His perspective towards life on many occasions have helped me deal with the pressures in IISc better. I am also thankful to Raj Rajveer, Sindhu Padakandla, Disha Dinesha, Monika Dhok, Srinivas Karthik and Rafia Sabih. Heartfelt thanks to my other friends Dr. Umang Mammaniya, Vrishali Shah, and Saurabh Shertukde for their constant encouragement and support during my PhD.

The past two years in IISc have been the best period of my life and this is mostly due to Vishakha Patil. She has been my partner in crime, food, entertainment, long walks, research discussions, and many more things. Thank you for sharing all the ups and the innumerable number of lows in my life and for always being there. I thank my sister and brother for their valuable advice and encouragement at different points in my PhD years. Finally I thank my parents whose contributions and sacrifices have ensured that I could pursue my dreams. Their support is unbounded and cannot be measured. I dedicate this thesis to them.
DEDICATED TO

MY PARENTS

Whose sacrifices have made this possible
Abstract

The iterated matrix multiplication polynomial ($IMM_{w,d}$) is the entry in the $1 \times 1$ matrix product $Q_1 \cdot Q_2 \cdots Q_d$, where $Q_1, Q_d$ are row and column matrices of length $w$, and $Q_i$ is a $w \times w$ matrix for $i \in [d]$, and the entries in these matrices are distinct variables. The number of variables in $IMM_{w,d}$ is equal to $n = w^2(d - 2) + 2w$. In this thesis, we study certain learning and lower bound problems related to $IMM_{w,d}$. Our first work gives an efficient equivalence test for $IMM_{w,d}$. The second work is related to learning affine projections of $IMM_{w,d}$ in the average-case, and our last work gives lower bounds on interesting restrictions of arithmetic formulas computing $IMM_{w,d}$.

An $n$ variate polynomial $f$ is equivalent to $IMM_{w,d}$ if there is an $A \in GL(n, F)$ such that $f = IMM_{w,d}(Ax)$. We design a randomized algorithm that given blackbox/oracle access to an $n$ variate, degree $d$ polynomial $f$, with high probability determines whether $f$ is equivalent to $IMM_{w,d}$. If $f$ is equivalent to $IMM_{w,d}$ then it returns a $w$ and an $A \in GL(n, F)$ such that $f = IMM_{w,d}(Ax)$. The algorithm works over $\mathbb{Q}$ and finite fields of large enough characteristic and runs in time $O((n \beta)^{O(1)})$, where $\beta$ is the bit length of the coefficients of $f$. The algorithm works even when $Q_i$ is $w_{i-1} \times w_i$ matrix for $i \in [d]$ ($w_0 = w_d = 1$), and $w_1, \ldots, w_{d-1}$ is unknown to the algorithm. At its core, the equivalence testing algorithm exploits a connection between the irreducible invariant subspaces of the Lie algebra of the group of symmetries of a polynomial $f$ that is equivalent to $IMM_{w,d}$ and the layer spaces of a full-rank algebraic branching program computing $f$. This connection also helps determine the group of symmetries of $IMM_{w,d}$ and show that $IMM_{w,d}$ is characterized by its group of symmetries.

An $m$ variate polynomial $f$ is computed by an algebraic branching program (ABP) of width $w$ and length $d$ if $f$ is an affine projection of $IMM_{w,d}$, that is there is an $A \in F^{n \times m}$ and a $b \in F^n$ such that $f = IMM_{w,d}(Ax + b)$. We study the average-case complexity of reconstructing ABPs under a natural distribution – the entries of $A$ and $b$ are chosen independently and uniformly at random. The equivalence testing algorithm for $IMM_{w,d}$ gives an efficient average-case reconstruction algorithm for width $w$ ABPs computing $m$ variate, degree $d$ polynomials when
Abstract

\[ w \leq \sqrt{\frac{m}{d}}. \]  We make progress on reducing this constraint on the width by studying a related problem called linear matrix factorization (LMF): the input is blackbox access to \( w^2, m \) variate degree \( d \) polynomials that are entries of the matrix product \( X_1 \cdots X_d \), where \( X_i \) is a \( w \times w \) matrix with linear polynomials as entries for \( i \in [d] \), and the outputs are \( d, w \times w \) matrices with linear polynomials as entries such that their product is equal to \( X_1 \cdots X_d \). We give a \( (md\beta)^O(1) \) time randomized algorithm that solves the average-case LMF problem when \( w \leq \sqrt{\frac{m}{2}} \). Here \( \beta \) is the bit length of the coefficients of the input polynomials. In fact, we give a polynomial time randomized algorithm that solves (worst-case) LMF problem when the input matrix product is non-degenerate or pure – a notion we define in this work. We show that the factorization of a pure product is unique in a certain sense, and at a high level it is this uniqueness that helps the algorithm to compute the factorization. Using the average-case LMF algorithm, we give a \( (d^w m \beta)^O(1) \) time algorithm for average-case ABP reconstruction when \( w \leq \sqrt{\frac{m}{2}} \), which is an interesting progress on learning ABPs in the context of the \( \frac{m}{2} \) width lower bound known for homogeneous ABPs [Kum17]. Both the algorithms work over \( \mathbb{Q} \) and finite fields of large enough characteristic, but over rationals our output matrices are over a degree \( w \) extension of \( \mathbb{Q} \).

On lower bounds, we prove a \( w^{\Omega(d)} \) size lower bound on multilinear depth three formulas computing \( \text{IMM}_{w,d} \). The lower bound is proved by introducing a novel variant of the partial derivatives measure called skewed partial derivatives, which found applications in other important subsequent works. Improving this result to a \( w^{\Omega(\log d)} \) size lower bound on general multilinear formulas computing \( \text{IMM}_{w,d} \) would imply a super-polynomial separation between ABPs and arithmetic formulas [Raz13], which is a long standing open problem. We also show an exponential separation between multilinear depth three and multilinear depth four formulas which was a substantial improvement over the quasi-polynomial separation already known [RY09]. We also consider a restriction of multilinear formulas, called interval set-multilinear formulas computing \( \text{IMM}_{w,d} \), where with every node an interval \( I \subseteq [1, d] \) is associated and the node computes a set-multilinear polynomial in the variables from the matrices \( Q_i, i \in I \). Further, the interval of a product node is the disjoint union of the intervals of its children and the interval of an addition node is equal to the intervals of its children. Proving a super-polynomial size lower bound on interval set-multilinear formulas computing \( \text{IMM}_{w,d} \) would imply a super-polynomial separation between non-commutative algebraic branching programs and non-commutative homogeneous formulas. We prove super-polynomial size lower bound on interval set-multilinear formulas computing \( \text{IMM}_{w,d} \), when the interval sizes of the product nodes are not too big compared to the intervals of its children.
Publications based on this Thesis

1. Average-case Linear Matrix Factorization and Reconstruction of Low Width Algebraic Branching Programs,
   Joint work with Neeraj Kayal and Chandan Saha,
   Accepted to Computational Complexity journal, 2019.

2. Reconstruction of Full-rank Algebraic Branching Programs,
   Joint work with Neeraj Kayal, Chandan Saha, and Sébastien Tavenas,
   Invited to ACM Transactions on Computation Theory (ToCT) journal, volume 11(1), 2018.

3. Separation Between Read-once Oblivious Algebraic Branching Programs (ROABPs) and Multilinear Depth Three Circuits,
   Joint work with Neeraj Kayal and Chandan Saha,
   Invited to ACM Transactions on Computation Theory (ToCT) journal, 2019.
   Conference version appeared in 33rd Symposium on Theoretical Aspects of Computer Science (STACS), 2016.
Contents

Acknowledgements 5
Abstract i
Publications based on this Thesis iii
Contents iv
List of Figures vii

1 Introduction 1
   1.1 Arithmetic circuit complexity 3
      1.1.1 Lower bounds 3
      1.1.2 Polynomial identity testing (PIT) 7
      1.1.3 Circuit reconstruction 10
      1.1.4 Equivalence testing 16
   1.2 Our contributions 18
      1.2.1 Equivalence test for IMM 18
      1.2.2 Average-case linear matrix factorization and low width ABP circuit reconstruction 20
      1.2.3 Lower bounds for IMM 26
      1.2.4 PIT for superposition of set-multilinear depth three circuits 28
   1.3 Organization 29

2 Preliminaries 30
   2.1 Algebraic circuit models 30
   2.2 Linear algebra 33
   2.3 Iterated matrix multiplication polynomial 34
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4</td>
<td>Complexity measures</td>
<td>34</td>
</tr>
<tr>
<td>2.5</td>
<td>Affine and $p$-projections</td>
<td>36</td>
</tr>
<tr>
<td>2.6</td>
<td>Group of symmetries and Lie algebra</td>
<td>37</td>
</tr>
<tr>
<td>2.7</td>
<td>Algorithmic preliminaries</td>
<td>39</td>
</tr>
<tr>
<td>3</td>
<td>Lie algebra of the IMM and the determinant polynomials</td>
<td>47</td>
</tr>
<tr>
<td>3.1</td>
<td>Lie algebra of IMM</td>
<td>47</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Structure of the Lie algebra $\mathfrak{g}_{\text{IMM}}$</td>
<td>47</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Random elements of $\mathfrak{g}_{\text{IMM}}$</td>
<td>59</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Invariant subspaces of $\mathfrak{g}_{\text{IMM}}$</td>
<td>60</td>
</tr>
<tr>
<td>3.2</td>
<td>Lie algebra of $\text{Det}_n$</td>
<td>64</td>
</tr>
<tr>
<td>4</td>
<td>Equivalence testing for IMM</td>
<td>71</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>71</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Algorithm and proof strategy</td>
<td>72</td>
</tr>
<tr>
<td>4.2</td>
<td>Almost set-multilinear ABP</td>
<td>77</td>
</tr>
<tr>
<td>4.3</td>
<td>Lie algebra of $f$ equivalent to IMM</td>
<td>79</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Computing invariant subspaces of the Lie algebra $\mathfrak{g}_f$</td>
<td>79</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Closure of a vector under the action of $\mathfrak{g}_f$</td>
<td>85</td>
</tr>
<tr>
<td>4.4</td>
<td>Reconstruction of full-rank ABP for $f$</td>
<td>86</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Computing layer spaces from invariant subspaces of $\mathfrak{g}_f$</td>
<td>87</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Reduction to almost set-multilinear ABP</td>
<td>90</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Reconstructing almost set-multilinear ABP</td>
<td>94</td>
</tr>
<tr>
<td>5</td>
<td>Symmetries of IMM</td>
<td>99</td>
</tr>
<tr>
<td>5.1</td>
<td>The group $\mathcal{G}_{\text{IMM}}$</td>
<td>99</td>
</tr>
<tr>
<td>5.2</td>
<td>Proof of Theorem 5.1</td>
<td>101</td>
</tr>
<tr>
<td>5.3</td>
<td>Characterization of IMM by $\mathcal{G}_{\text{IMM}}$</td>
<td>104</td>
</tr>
<tr>
<td>6</td>
<td>Average-case LMF and reconstruction of low width ABP</td>
<td>107</td>
</tr>
<tr>
<td>6.1</td>
<td>Introduction</td>
<td>107</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Algorithms and their analysis</td>
<td>109</td>
</tr>
<tr>
<td>6.1.1.1</td>
<td>Analysis of Algorithm 10</td>
<td>109</td>
</tr>
<tr>
<td>6.1.1.2</td>
<td>Analysis of Algorithm 11</td>
<td>112</td>
</tr>
<tr>
<td>6.1.1.3</td>
<td>Analysis of Algorithm 12</td>
<td>115</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Dependencies between the algorithms</td>
<td>116</td>
</tr>
<tr>
<td>Section</td>
<td>Pages</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>6.2 Purity of random matrix product</td>
<td>117</td>
<td></td>
</tr>
<tr>
<td>6.3 Average-case linear matrix factorization: Proof of Theorem 1.2</td>
<td>118</td>
<td></td>
</tr>
<tr>
<td>6.3.1 Rearranging the matrices</td>
<td>118</td>
<td></td>
</tr>
<tr>
<td>6.3.2 Determining the last matrix: Proof of Lemma 6.1</td>
<td>120</td>
<td></td>
</tr>
<tr>
<td>6.4 Average-case ABP reconstruction: Proof of Theorem 1.3</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>6.4.1 Computing the corner spaces</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>6.4.2 Finding the coefficients in the intermediate matrices</td>
<td>135</td>
<td></td>
</tr>
<tr>
<td>6.4.3 Non-degenerate ABP</td>
<td>141</td>
<td></td>
</tr>
<tr>
<td>6.5 Equivalence test for the determinant over finite fields</td>
<td>142</td>
<td></td>
</tr>
<tr>
<td>6.5.1 Random element in the Lie algebra of determinant</td>
<td>143</td>
<td></td>
</tr>
<tr>
<td>6.5.2 Reduction to PS-equivalence testing</td>
<td>146</td>
<td></td>
</tr>
<tr>
<td>7 Lower bounds for IMM</td>
<td>148</td>
<td></td>
</tr>
<tr>
<td>7.1 Introduction</td>
<td>148</td>
<td></td>
</tr>
<tr>
<td>7.1.1 Proof strategy for Theorems 1.4 and 1.7</td>
<td>149</td>
<td></td>
</tr>
<tr>
<td>7.2 Lower bound on multilinear depth three formulas</td>
<td>150</td>
<td></td>
</tr>
<tr>
<td>7.2.1 Proof of Theorem 1.4</td>
<td>151</td>
<td></td>
</tr>
<tr>
<td>7.2.2 Proof of Theorem 1.5</td>
<td>153</td>
<td></td>
</tr>
<tr>
<td>7.3 Lower bound on $\alpha$-balanced ISM formulas</td>
<td>153</td>
<td></td>
</tr>
<tr>
<td>7.3.1 Reduction from homogeneous non-commutative formulas to ISM formulas</td>
<td>153</td>
<td></td>
</tr>
<tr>
<td>7.3.2 Proof of Theorem 1.7</td>
<td>155</td>
<td></td>
</tr>
<tr>
<td>8 Future Work</td>
<td>159</td>
<td></td>
</tr>
<tr>
<td>Bibliography</td>
<td>162</td>
<td></td>
</tr>
</tbody>
</table>
List of Figures

2.1 Naming of variables in IMM_{w,d} ............................................. 34
3.1 A matrix \( E \) in \( \mathcal{g}_{\text{MM}} \) ............................................................... 49
3.2 An ABP computing the term \( f_{ij}^{(k)} \cdot \frac{\partial \text{IMM}}{\partial x_{ij}^{(k)}} \) .................................................. 51
3.3 A matrix \( C^{(a)} \) in \( \mathcal{W}_3^{(a)} \) .......................................................... 52
3.4 An ABP computing the term \( x_p^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(2)}} \) ...................................... 53
3.5 An ABP computing the term \( x_q^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(2)}} \) ...................................... 53
3.6 A matrix \( C^{(a)} \) in \( \mathcal{W}_3^{(a)} \) .......................................................... 54
3.7 Submatrix of \( L \) restricted to rows/columns indexed by \( x_k \) ................. 62
3.8 Submatrix of \( M \) matrix restricted to rows/columns indexed by \( x_1 \uplus x_2 \) ...... 63
4.1 Random element \( R \) in \( \mathcal{g}_{\text{MM}} \) ............................................................... 80
4.2 Matrix \( h_k(R) \) .......................................................... 82
4.3 Matrices \( E_1, E_d \) and \( E_k \) .................................................. 88
4.4 Matrices \( E \) and \( E^{-1} \) .................................................. 89
5.1 Matrix \( A \) in \( \mathcal{G}_{\text{MM}} \) ............................................................... 102
6.1 Dependencies between the three algorithms ............................................ 117
7.1 ABP \( \mathcal{M} \) ............................................................... 151
Chapter 1

Introduction

Algorithms for algebraic problems have been studied since time immemorial and many beautiful algorithms have been proposed for these problems, for example the Newton’s classical iterative method for finding roots of a univariate polynomial, a fast algorithm to compute discrete Fourier transform made popular by Cooley and Tukey [CT65], algorithms by Strassen and those after him for matrix multiplication [Str69, CW90, Sto10, Wil14, Gal14] (also see [PBS10] for a detailed survey on matrix multiplication), and parallel algorithms to compute the determinant of a matrix [Csa76, Ber84, Chi85] are few among many others. Many of these algorithms perform computations over real or complex numbers and the output of these algorithms is a polynomial function in the inputs, for example by Leibniz formula, the determinant of a matrix is a polynomial in the entries of the matrix. Moreover, if we consider the entries of the matrix to be formal variables then the algorithms to compute the determinant only perform basic arithmetic operations like addition, subtraction, multiplication or division with rational polynomials in the input variables as the operands. Hence, to analyse the complexity of these algorithms it is natural to desire a computational model that abstracts out the bit level computations by allowing the algorithm to perform basic operations like addition/multiplication with field elements free of cost. Additionally such a model should also enable the algorithm to perform arithmetic operations over the rational functions in the input variables at unit cost. Inspired by the Turing machine model, Lenore Blum, Michael Shub and Steve Smale defined an algebraic model called the Blum-Shub-Smale (BSS) model which encapsulates these two properties and aids in better analysing the computational complexity.

1The algorithm was first proposed by Gauss and later on it was rediscovered many times in various limited forms but Cooley and Tukey independently came up with the algorithm and also analysed its asymptotic complexity.

2Think of Gaussian elimination.
of algebraic algorithms.

The BSS model is a uniform model i.e. there is one algorithm for inputs of all sizes, and its non-uniform analogs are the algebraic computation trees and the arithmetic circuits/straight line programs\textsuperscript{1}. Both these algebraic models were considered before the BSS model was defined\textsuperscript{2} and in fact, the BSS model used the ideas these models were based upon to include the concept of uniformity in algebraic computation. We restrict our attention to the arithmetic circuit model in this work and refer the reader to \cite{BSS88} and \cite{PBS10} for more on the BSS model and algebraic computation tree respectively. Arithmetic circuits are formally defined in Definition 2.1. The input to an arithmetic circuit are the \( x = \{x_1, \ldots, x_n\} \) variables and it computes a polynomial in \( x \) variables using the four basic arithmetic operations\textsuperscript{3}. An arithmetic circuit can be viewed as an algorithm computing a polynomial function in \( x \) variables. The two complexity measures associated with an arithmetic circuit are its \textit{size} and \textit{depth}. The size is equal to the number of operations performed by the arithmetic circuit and corresponds to the sequential time required by the circuit to compute the polynomial, whereas depth is equal to the parallel time required by the circuit to compute the polynomial. A fundamental problem in arithmetic circuit complexity is to prove explicit size/depth lower bounds on arithmetic circuits. In Section 1.1, we elaborate on this lower bound problem and its surprising connections with other algorithmic problems in algebraic complexity theory like polynomial identity testing, circuit reconstruction and equivalence testing. In order to put our results in context, in Section 1.1 we also give a brief account of the previous results on lower bound, polynomial identity testing, reconstruction and equivalence testing. Then in Section 1.2, we state our contributions on each of these problems, and we conclude this chapter by providing the outline for the organization of this thesis in Section 1.3.

\textsuperscript{1}This is similar to the Boolean circuit model which is the non-uniform model corresponding to the Turing machine.

\textsuperscript{2}Arithmetic circuits were first considered in the work of \cite{Ost54} whereas the first definitions of algebraic computation trees appear in the works of \cite{Str72, Rab72}.

\textsuperscript{3}In Definition 2.1 an arithmetic circuit computes a polynomial using only addition and multiplication operations. The absence of division operation is justified by a result due to Strassen \cite{Str73b} which we state later, whereas subtraction can be simulated by multiplying with \(-1\).
1.1 Arithmetic circuit complexity

1.1.1 Lower bounds

**Algebraic complexity classes**: Similar to Boolean function families being classified into complexity classes, in the algebraic world polynomial families are classified into algebraic complexity classes based on how efficiently they can be computed using arithmetic circuits. In this context, efficiency refers to the size (or depth) of an arithmetic circuit. Analogous to $P$ and $NP$ in the Boolean world, Valiant [Val79a, Val79b] defined the algebraic complexity classes $VP$ and $VNP$. A polynomial family $\{f_n\}_{n \in \mathbb{N}}$, where $f_n$ is an $n$ variate polynomial, belongs to $VP$ if the degree of $f_n$ is $n^{O(1)}$ and $f_n$ is computable by an $n^{O(1)}$ size arithmetic circuit, for all $n \in \mathbb{N}$. On the other hand $\{f_n\}_{n \in \mathbb{N}}$ is in $VNP$ if there is a polynomial family $\{g_m\}_{m \in \mathbb{N}}$ in $VP$ and a polynomial function $t : \mathbb{N} \to \mathbb{N}$ such that for every $n \in \mathbb{N}$ there is an $m \leq t(n)$ satisfying

$$f_n(x) = \sum_{y \in \{0,1\}^{m-n}} g_m(x, y).$$

To understand the motivations for considering polynomial degree restriction on the polynomial families in $VP$ and $VNP$ we refer the reader to the answer given by Joshua Growchow in [Gro13]. An important result by Strassen [Str73b] shows that if there is an arithmetic circuit (having addition, division and multiplication gates) of size $s$ computing an $n$ variate degree $d$ polynomial $f$ then there is a circuit of size $(sd)^{O(1)}$ that computes $f$ and performs no division operations. Since the degrees of interesting polynomial families, like those in $VP$ and $VNP$ are polynomially bounded and our primary interest lies in determining whether these polynomial families are computable by polynomial size circuits, we henceforth assume that no division operations are performed by an arithmetic circuit.

Observe that $VP \subseteq VNP$, and Valiant conjectured that $VP \neq VNP$. Till today this remains one of the most important unresolved conjectures in this area. Well-known examples of polynomial families in $VP$ are the elementary symmetric family, the power symmetric family, the determinant family and the iterated matrix multiplication ($IMM$) family (see Section 2.3). Valiant [Val79a] also showed that over fields of characteristic not equal to two, the family of permanent polynomials is $VNP$-complete under $p$-projections (see Definition 2.15). Subsequently other polynomial families derived from computational problems related to graphs, like computing the number of cliques or Hamiltonian cycles in a directed graph were shown to be $VNP$-complete [B00].
There are two other important circuit models studied in arithmetic circuit complexity, namely algebraic branching programs (ABP) (see Definition 2.4) and arithmetic formulas (see Definition 2.2). A polynomial family \( \{f_n\}_{n \in \mathbb{N}} \) is in the complexity class \( \text{VBP} \) (respectively \( \text{VF} \)) if \( f_n \) is computable by an \( n^{O(1)} \) size ABP (respectively arithmetic formula). The determinant and IMM polynomial families are complete for \( \text{VBP} \) under \( p \)-projections [MV97], whereas width three IMM polynomial family is complete for \( \text{VF} \) under \( p \)-projections [BC92]. It is easy to infer from the definitions of ABP and arithmetic formulas that \( \text{VF} \subseteq \text{VBP} \subseteq \text{VP} \subseteq \text{VNP} \).

To separate any of these complexity classes remains a major open problem. In the past fifty years, a significant part of the research in this area has focused on proving super-polynomial lower bounds on mostly restricted arithmetic circuits/ABPs/formulas, and in the next section we state few of the well-known results among them. From here on we would refer to the operations performed by a circuit as a node/gate in the circuit, as each node/gate in the circuit corresponds to an operation.

**Previous work**

**General arithmetic circuit/ABP/formula:** Owing to the structure in the computations performed by arithmetic circuit models, proving lower bounds for them is believed to be easier than proving lower bounds on Boolean circuits, and there is hope that we might be able to use well-developed tools from different branches of mathematics to prove lower bounds in the algebraic setting. An indication of this is that \( \text{P/poly} \neq \text{NP/poly} \) implies \( \text{VP} \neq \text{VNP} \) under the assumption that generalized Reimann hypothesis is true [B00]. Unfortunately though, we do not know of any super-polynomial lower bound for arithmetic circuits, ABPs or formulas. The best known size lower bound for arithmetic circuits or ABPs\(^1\) is \( \Omega(n \log d) \) due to [BS83, Str73a] where the hard polynomial is \( x_1^d + \ldots + x_n^d \). A crucial ingredient in the lower bound proof of [BS83, Str73a] is the following structural result: if there is a size \( s \) arithmetic circuit computing a polynomial \( f \) then there is an \( O(s) \) size arithmetic circuit that simultaneously computes the first order partial derivatives of \( f \). The lower bound is then proved by using Bezout’s theorem to show that any arithmetic circuit simultaneously computing the polynomials \( x_1^d, \ldots, x_n^d \) requires size at least \( \Omega(n \log d) \). For arithmetic formulas though, using a complexity measure based on the transcendental degree of a set of polynomials, an \( \Omega(n^3) \) size lower bound is known due to [Kal85] where the hard polynomial is \( \text{Det}_n \) – the

---

\(^1\)The lower bound for ABPs is on the number of edges in the ABP.
determinant of the $n \times n$ symbolic matrix\(^1\). Recently [Kum17] proved an $\Omega(nd)$ size lower bound on homogeneous ABPs\(^2\) where the hard polynomial is $x_1^d + \ldots + x_n^d$. [Kum17] actually proved that the width of every layer in a homogeneous ABP computing $x_1^d + \ldots + x_n^d$ is at least $n^\frac{1}{d}$, and hence the number of nodes in the ABP is $\Omega(nd)$. The argument in [Kum17] is based on a lower bound on the dimension of the variety of a set of homogeneous polynomials.

**Depth-reduction results:** Unlike Boolean circuits, a super-polynomial lower bound is not known for even constant depth arithmetic circuits. To prove a super-polynomial lower bound on depth $k$ arithmetic circuits, we can assume that the circuit constitutes of $k$ alternating layers of addition and multiplication gates. This assumption holds without loss of generality as there is an easy reduction to convert any depth $k$ arithmetic circuit to such an alternating layered circuit with only a polynomial blow-up in the size. Further, if the hard polynomial is irreducible as in most cases, then we can also assume that the root or the output gate of the circuit is an addition gate. Thus the simplest circuit model is the depth two model denoted $\Sigma \Pi$ circuits. From the perspective of lower bounds though, this is not an interesting model as the number of monomials in a polynomial is a lower bound on the size of $\Sigma \Pi$ circuit computing it. Thus the simplest but still non-trivial arithmetic circuit model is the $\Sigma \Pi \Sigma$ arithmetic circuit, and even for this model there is no super-polynomial lower bound known\(^3\). An almost cubic lower bound is the best known for $\Sigma \Pi \Sigma$ circuits [KST16a, SW01]. A partial justification for the absence of strong lower bound for $\Sigma \Pi \Sigma$ circuits is given via a series of depth reduction results. Firstly, in a remarkable result by [VSBR83] it was shown that if an $n$ variate degree $d$ polynomial is computable by a size $s$ arithmetic circuit then there is a size $(nds)^{O(1)}$ and depth $O((\log d)(\log d + \log s))$ fan-in two arithmetic circuit computing $f$, and hence $VP = VNC_2$ \(^4\). Building on this depth reduction result, [AV08, Koi12, GKKS16, Tav15] showed that if an $n$ variate degree $d = n^{O(1)}$ polynomial is computable by a size $s$ arithmetic circuit then it is computed by a size $s^{O(\sqrt{d})}$ homogeneous $\Sigma \Pi \Sigma \Pi$ circuit and over characteristic zero fields by a size $s^{O(\sqrt{d})}$ (non-homogeneous) $\Sigma \Pi \Sigma$ arithmetic circuit. Thus, an $n^{\omega(\sqrt{d})}$ size lower bound on either $\Sigma \Pi \Sigma$ or homogeneous $\Sigma \Pi \Sigma \Pi$ circuits computing an $n$ variate degree $d$ polynomial in $VNP$ would be sufficient to prove $VP \neq VNP$. A few lower bound results have come close to achieving this goal. An $n^{\Omega(\sqrt{d})}$ size lower bound on homogeneous

---

\(^1\)Notice that this is a sub-quadratic lower bound, but the same analysis can be used to show a quadratic lower bound for the polynomial $\sum_{i=1}^{n} \sum_{j=1}^{n} x_i^j y_j$ (survey by [Sap15]).

\(^2\)Every node in a homogeneous ABP computes a homogeneous polynomial.

\(^3\)Exponential lower bound for $\Sigma \Pi \Sigma$ circuits are known over finite fields [GK98], and exponential lower bounds are also known for homogeneous $\Sigma \Pi \Sigma$ circuits over any field [NW97].

\(^4\)A family of polynomials $\{f_n\}_{n \in \mathbb{N}}$ is in $VNC_2$ if degree of $f_n$ is $n^{O(1)}$ and $f_n$ is computable by a size $n^{O(1)}$ and depth $O((\log n)^k)$ fan-in two arithmetic circuit, for all $n \in \mathbb{N}$. 


circuits computing the Nisan-Wigderson\textsuperscript{1} and the IMM polynomials were shown in [KSS14, FLMS15, KLSS17, KS17].

**Multilinear models:** Given the state of art, it is natural to wonder whether we can prove lower bounds on further restricted circuit models. Multilinearity is one such natural restriction where a multilinear polynomial is computed at every node in a circuit/ABP/formula. There are at least three motivations to consider multilinear circuit models: a) Important polynomials like the determinant, permanent and IMM are multilinear and it is a natural question to determine the complexity of computing such polynomials using multilinear circuit models, b) a super-polynomial size lower bound on multilinear formulas computing an $n$ variate multilinear polynomial $f$ of degree $O\left(\frac{\log n}{\log \log n}\right)$ implies a super-polynomial size lower bound on arithmetic formulas computing $f$ [Raz13], and c) techniques developed to prove lower bounds for them might give us insights to prove lower bounds for general arithmetic circuits. There is a large body of work devoted to proving lower bounds for multilinear circuit models and among them the most outstanding result is the quasi-polynomial lower bound on the size of multilinear formulas computing the determinant or the permanent by [Raz09], which is obtained by ingeniously analysing the rank of the partial derivative matrix of a polynomial computed by a multilinear formula. Building on the lower bound by [Raz09], [Raz06] and [RY08] showed a similar lower bound on the size of multilinear formulas computing an explicit polynomial that can be computed by a polynomial size multilinear circuit. This result in particular showed a quasi-polynomial separation between multilinear circuits and multilinear formulas. Later in [DMPY12], a quasi-polynomial separation between multilinear ABPs and multilinear formulas was shown by proving a quasi-polynomial lower bound on the size of multilinear formulas computing an explicit polynomial (namely an arc full-rank polynomial) that can be computed by a polynomial size multilinear circuit. [RY09] also showed a super-polynomial separation between multilinear formulas of constant product-depth $\Delta$ and $\Delta + 1$.

In this work we prove a $2^{\Omega(n)}$ separation between multilinear formulas of product depths \(\Delta\) and \(\Delta + 1\) (see Theorem 1.5). In [CELS18] this was improved to an optimal $2^{\Omega(n^{\frac{1}{\Delta}})}$ separation between multilinear formulas of product-depth $\Delta$ and $\Delta + 1$ when $\Delta = o\left(\frac{\log n}{\log \log n}\right)$. Further [CLS19] showed that over characteristic zero fields, there is a polynomial computable by a size $s$ arithmetic formula of product depth $\Delta$ which cannot be computed by a size $s^{o(1)}$ size multilinear formula of product depth $\Delta$, when $\Delta = o(\log s)$. Improving the restriction of $\Delta$ in [CLS19] to $O(\log s)$ will imply a super-polynomial separation between arithmetic formulas and multilinear formulas [Bre74].

\textsuperscript{1}a polynomial family in VNP
Lower bounds for IMM: Since $\text{IMM}_{w,d}$ is complete for ABPs under $p$-projections, proving $\text{VBP} \neq \text{VF}$ is equivalent to showing a super-polynomial lower bound on the size of arithmetic formulas computing $\text{IMM}_{w,d}$. Using the divide and conquer approach one can easily show that there is an arithmetic formula of size $w^{O(\log d)}$ and product-depth $\log d$ computing $\text{IMM}_{w,d}$. This recursive formula is in fact multilinear, and no multilinear formula or even arithmetic formula computing $\text{IMM}_{w,d}$ of smaller size is known. A non-trivial way to transform an arithmetic formula into a multilinear formula by [Raz13] implies that, to prove $\text{VBP} \neq \text{VF}$ it suffices to show either of the following: a) a super-polynomial size lower bound on multilinear formulas computing $\text{IMM}_{w,d}$ when the degree $d = O(\log \log w)$, or b) a $w^{\Omega(\log d)}$ size lower bound on multilinear formulas computing $\text{IMM}_{w,d}$ for $d = w^{\omega(\log d')}$, where $d' = O\left(\frac{\log w}{\log \log w}\right)$.

Further, even proving a super-polynomial size lower bound on certain restricted multilinear formulas called interval set-multilinear formulas (see Definition 1.2) would imply a separation between homogeneous formulas and ABPs in the non-commutative setting.

Although $\text{Det}_n$ is a $p$-projection of $\text{IMM}_{w,d}$ where $w$ and $d$ are polynomially related to $n$ [MV97], the quasi-polynomial size lower bound proved on multilinear formulas computing $\text{Det}_n$ in [Raz09] does not imply the same for multilinear formulas computing $\text{IMM}_{w,d}$. This is because the projection is not multilinear, i.e. under the projection, two variables from $\text{IMM}_{w,d}$ map to the same variable in $\text{Det}_n$ and this destroys the multilinearity of the formula. A similar thing is also true for the hard polynomial (arc full-rank polynomial) in [DMPY12]. In this work we prove a $w^{\Omega(d)}$ size lower bound on multilinear $\Sigma \Pi \Sigma$ formulas computing $\text{IMM}_{w,d}$ (see Theorem 1.4). Subsequent to our work [KST16b] proved a $w^{\Omega(\sqrt{d})}$ size lower bound on multilinear depth four formulas computing $\text{IMM}_{w,d}$ by significantly extending the techniques of our work. Recently, a $2^{\Omega(\Delta d^{1.5})}$ size lower bound on product depth $\Delta$ multilinear formulas computing $\text{IMM}_{2,d}$ was shown when $\Delta \leq \log d$ in [CLS19] building upon a similar lower bound for set-multilinear formulas in [NW97].

1.1.2 Polynomial identity testing (PIT)

Problem definition and connection to lower bounds: PIT is an algorithmic problem where a polynomial $f$ computed by a circuit $C$ is given as input to the algorithm, and the task is to efficiently determine whether the polynomial computed by $C$ is non-zero. Here, we

---

[1] We thank Srikanth Srinivasan for bringing implication b to our notice. We refer the reader to Chapter 7 after Theorem 1.4 is restated for a short justification of this implication.

[2] The variables in a non-commutative polynomial do not commute under multiplication. Thus $x_1 x_2 \neq x_2 x_1$ as non-commutative polynomials.
say an algorithm is efficient if it performs the above task in time polynomial in the size of $C$ and the degree of $f$. The problem is studied in two flavours: a) blackbox/oracle access to $C$, where the polynomial can only be queried at points from the field, and b) whitebox access to $C$, where the directed acyclic graph corresponding to $C$ is given as an input. An easy blackbox randomized algorithm for PIT follows from Schwartz-Zippel lemma [DL78, Zip79, Sch80] but no sub-exponential time algorithm is known for even whitebox PIT. The PIT problem is one of the very few natural problems known to be in BPP but not known to be in P. A major reason for this is the connection between PIT and circuit lower bounds. In [KI04] it was shown that a polynomial time algorithm for whitebox PIT over integers implies either super-polynomial lower bounds for the family of permanent polynomial or $\text{NEXP} \not\subset \text{P/poly}$. Also [HS80] and later [Agr05] showed that a polynomial time blackbox PIT implies exponential lower bounds for circuits computing polynomials whose coefficients can be computed in PSPACE. In the other direction, [KI04] showed that a super-polynomial lower bound on any exponential time computable multilinear polynomial family implies a quasi-polynomial time blackbox PIT. A similar connection between PIT and lower bounds as in [KI04] was later shown for low depth circuits [DSY09]. Algorithms for PIT from lower bounds have further been explored in the recent work of [KSS19]. Apart from its connection to lower bounds, efficient derandomization of PIT is interesting in its own right as it has several applications like parallel algorithms for perfect matching [Lov79, KUW86, MVV87] and primality testing [AB03]. In fact the polynomial time algorithm for primality testing [AKS02] and the quasi-NC algorithm for perfect matching were obtained by derandomizing special instances of the PIT problem [FGT16, ST17]. Since designing an efficient deterministic PIT algorithm is a difficult problem, it has been accomplished for very few circuit models. Moreover, although PIT and lower bounds are seemingly equivalent problems for general circuits, for special circuit models it is usually believed from experience that proving lower bounds for them is easier than designing a PIT algorithm. Nevertheless the techniques used to prove a lower bound for restricted circuit models might shed some light on how to design PIT algorithms for related models.

**Previous work**

The simplest circuit model for PIT is a $\Sigma \Pi$ circuit for which a polynomial time blackbox PIT is known [BT88, KS01]. From the depth reduction results stated in the previous section and the connection between PIT and lower bounds stated above, it follows that a polynomial time PIT for $\Sigma \Pi \Sigma$ or homogeneous $\Sigma \Pi \Sigma \Pi$ circuit implies a quasi-polynomial time PIT for general

---

1. which also implies $\text{VP} \neq \text{VNP}$
arithmetic circuits computing low-degree polynomials (see Theorem 5.3 in [GKKS16]). Since depth three or depth four PIT is a seemingly hard problem, the next step with the view of making progress would be to study PIT for special types of these circuits. A depth three powering circuit computes a polynomial which is a sum of powers of linear forms (denoted $\Sigma \land \Sigma$), that is the product gates in the $\Sigma\Pi\Sigma$ circuits are replaced by powering gates, and [Sax08] used the duality trick to express such a polynomial as a sum of products of univariate polynomials for which a polynomial time whitebox PIT was known due to [RS05]. An $n^{O(\log \log k)}$ time blackbox PIT is known for depth three powering circuit where $k$ is the fan-in of the root node (henceforth called top fan-in) [FSS14]. Following a long sequence of works, a polynomial time blackbox PIT algorithm is now known for $\Sigma\Pi\Sigma$ circuits when the top fan-in is a constant [DS07, KS07, KS09b, KS11, SS11, SS12, SS13]. Later using the Jacobian, [ASSS16] gave a polynomial time blackbox PIT for $\Sigma\Pi\Sigma$ formulas when the set of polynomials computed by the product gates at layer two have a constant transcendence degree. [ASSS16] also gave a polynomial time blackbox PIT for constant occur formulas with constant depth, which in particular subsumes constant read formulas of constant depth and $\Sigma\Pi\Sigma\Pi$ multilinear formulas with constant top fan-in for which polynomial time blackbox PIT algorithms were given in [AvMV11, SV18]. A quasi-polynomial time blackbox PIT for constant read multilinear formulas without any depth restriction was also given in [AvMV11]. Recently [MV18] gave a polynomial time blackbox PIT for read-once formulas with no restriction on depth. [For15] used the shifted partial measure to give a quasi-polynomial time blackbox PIT algorithm for polynomials which are sums of powers of constant degree polynomials ($\Sigma \land \Sigma\Pi$ circuits, where the fan-in of the product gates at the fourth layer is a constant).

Since lower bounds for multilinear depth three circuits are known due to [CELS18, RY08] and also from this work, there is hope that an efficient PIT algorithm can be designed for multilinear depth three circuits. A sub-exponential time blackbox PIT is known for multilinear depth three circuits [dOSlV16] but no quasi-polynomial time blackbox PIT is known for this model. Multilinear depth three circuit computes a polynomial which is a sum of products of linear polynomials where each product term partitions the variables among the linear polynomials. A set-multilinear depth three circuit is a multilinear depth three circuit where the partition across the product terms remains the same. A polynomial time whitebox PIT algorithm for set-multilinear depth three circuit follows from [RS05], and later in [FS12] a quasi-polynomial time blackbox PIT was given when the partition among the variables in the product gates are known. Finally in [ASS13] using the shift and rank concentration technique a truly blackbox algorithm running in quasi-polynomial time was designed for set-multilinear
depth three circuits, where even the partition is unknown. The model read-once oblivious
ABP (ROABP) is a special type of algebraic branching program where each layer is associated
with a unique variable and an edge is labelled by a univariate polynomial in the variable asso-
ciated with the layer the edge belongs to. Computationally, ROABPs subsume set-multilinear
depth three circuits, that is if a polynomial $f$ is computable by a size $s$ set-multilinear depth
three circuit then $f$ is also computable by a size $s^{O(1)}$ ROABP. Again for ROABPs a poly-
momial time whitebox PIT follows from [RS05] and later a quasi-polynomial time blackbox PIT
with known variable ordering was given in [FS13]. Using the technique of basis isolation a
quasi-polynomial time blackbox PIT with unknown variable ordering was given in [AGKS15].

1.1.3 Circuit reconstruction

Problem definition and hardness: Reconstruction of arithmetic circuits is the algebraic ana-
logue of exact learning [Ang88] of Boolean circuits. A reconstruction algorithm takes as input
a blackbox access to an $n$ variate degree $d$ polynomial $f$ computed by a size $s$ arithmetic cir-
cuit from some circuit model$^1$ $C$, and outputs an arithmetic circuit (preferably from the same
model) of size not much larger than $s$ (ideally, a polynomial or quasi-polynomial function of
$s$) computing $f$. Thus, hardness of reconstruction is related to hardness of approximation
of the minimum circuit size problem. The hardness of the minimum circuit size problem
for Boolean circuits, known as MCSP, is an intensely studied problem in the literature. In
the MCSP problem, we are given the truth-table of a Boolean function and a parameter $s$
as input and the task is to determine if the function can be computed by a Boolean circuit
of size at most $s$. Allender and Hirahara [AH17] showed that approximating the minimum
circuit size within a factor of $N^{1-o(1)}$ is NP-intermediate, assuming the existence of one-way
functions, where $N$ is the size of the input truth-table. Another related result is the hardness
of approximating minimum size DNF. Umans [Uma99, TUZ07] showed that $n^{1-\epsilon}$ factor ap-
proximation of the minimum DNF size of an input DNF of size $n$, for every constant $\epsilon \in (0, 1)$,
is $\Sigma^p_2$-hard. Drawing analogy between the Boolean and the arithmetic worlds, we expect the
reconstruction problem to be hard even if the polynomial function $f$ is given verbosely as
a list of coefficients, and it only gets harder if $f$ is given as a blackbox. If we insist on a
very small approximation factor and on computing an output circuit that belongs to the same
model $C$ as the input circuit (as in proper learning), then the problem becomes NP-hard even
for simple circuit models like set-multilinear depth three circuits and depth three powering
circuits [BIJL18, SWZ19, Shi16, Hås90].

$^1$For example an ABP or an ROAB, in which case we say $f$ is computable by a size $s$ ABP or ROAB
respectively.
Connection to lower bounds: It is also known that efficient reconstruction implies lower bounds. It was shown in [FK09] that a randomized polynomial time reconstruction algorithm for an arithmetic circuit model $C$ implies the existence of a function in $\text{BPEXP}$ that does not have polynomial size circuits from $C$. Also, Volkovich [Vol16] showed that a deterministic polynomial time reconstruction algorithm for $C$ implies the existence of a polynomial $f$ computable in $2^{O(n)}$ time such that any circuit from $C$ computing $f$ has exponential size. Moreover, an efficient reconstruction algorithm for a circuit model automatically gives an efficient blackbox identity testing algorithm for the same model. In this sense reconstruction for a circuit model is a harder problem than lower bounds or PIT, and not much is known about the reverse direction: Do strong lower bounds for a circuit model imply efficient reconstruction for the same model? Even if we believe in the existence of explicit polynomials with high circuit complexity, we may not hope to get such an implication unconditionally as reconstruction seems to be an inherently hard problem. However, the answer is less clear for lower bound proofs with additional features such as “natural proofs”. Taking inspiration from natural proofs defined in [RR97], the notion of algebraic natural proofs is defined in [FSV17, GKSS17] to explore the limitations of existing techniques in proving $\text{VP} \neq \text{VNP}$.

Does natural lower bound proofs imply reconstruction? The intuitive reason for expecting a somewhat positive answer rests on the high level view that a natural lower bound proof (in the sense of [RR97]) is able to “efficiently” check some property of polynomials computed by a circuit model, and the same property might be potentially useful in designing reconstruction algorithms for the model. Indeed, for Boolean circuits, an interesting result [CIKK16] showed that the natural lower bound proof framework [RR97] for $\text{AC}^0[p]$ circuits can be used to give a quasi-polynomial time PAC learning under the uniform distribution and with membership queries for the same model. The result generalizes to any circuit model $C$ containing $\text{AC}^0[p]$ for some prime $p$, the “usefulness” parameter of a natural proof for $C$ determines the efficiency of such a PAC learning algorithm for $C$. This generic result is preceded by evidences that hinted at such a connection, like the learning algorithms for $\text{AC}^0$ circuits [LMN93] and $\text{AC}^0$ circuits with few majority gates [JKS02]. For circuit models whose known lower bound proofs do not fit in the natural proof framework, the situation is less clear. Examples of such models are $\text{ACC}^0$ [Wil14] and monotone circuits [Raz85]. A hardness result for polynomial-time learning of monotone circuits is known assuming the existence of one-way functions.

\textsuperscript{1}In another interesting work [EGdOW18], limitations of rank based lower bound methods have been shown \textit{unconditionally} towards achieving strong lower bounds for set-multilinear $\Sigma\Pi\Sigma$ circuits and $\Sigma \wedge \Pi$ circuits.
Analogous to Boolean circuits, does an algebraically natural lower bound proof (in the sense of [FSV17, GKSS17]) for an arithmetic circuit model imply efficient reconstruction for the same model? Unlike PAC learning, in the algebraic setting we need to reconstruct a circuit that computes the input polynomial exactly instead of approximately, as two distinct polynomial functions differ at far too many points. If we insist on such exact learning in the Boolean setting (which is closely related to the compression problem for Boolean functions) then the best known output circuit size for $\text{AC}^0$ and $\text{AC}^0[p]$ functions is exponential in the number of variables [CKK+15, Sri15, CIKK16], and the best known algorithm for learning size $s$ DNF in $n$ variables has time complexity $2^{O(n^{3/2} \log n \log s)}$ [KS04]. In the absence of a generic connection (analogous to [CIKK16]) in the algebraic setting, we could gather more evidences for or against such a connection by focusing on restricted classes for which natural lower bound proofs are known.

There are many interesting arithmetic circuit models for which we know of strong lower bounds (that are also algebraically natural), but not efficient reconstruction algorithms. Instances of such models are homogeneous depth three circuits [NW97, KST16a], homogeneous depth four circuits [KLSS17, KS17], constant depth multilinear circuits [RY09, CLS19], multilinear formulas [Raz09], regular formulas [KSS14], and a few other classes [KS16a, KS16b]. Even for a more powerful model like homogeneous ABPs, it makes sense to ask – can we reconstruct sub-linear width homogeneous ABPs efficiently? Recall that a linear width lower bound for homogeneous ABPs is known [Kum17], and this lower bound proof is also natural. Unfortunately, there is some amount of evidence that indicate that the problem remains hard in the worst-case even for models for which natural lower bound proofs are known. For example, a polynomial time reconstruction algorithm for homogeneous $\Sigma \Pi \Sigma$ circuits with high degree implies the same for (non-homogeneous) $\Sigma \Pi \Sigma$ circuits and hence, also a sub-exponential time reconstruction algorithm for general circuits due to the depth reduction results. Such a reconstruction algorithm would also give a super-polynomial lower bound for $\Sigma \Pi \Sigma$ circuits via the learning to lower bound connection [FK09]. Similarly, a polynomial time reconstruction algorithm for constant width (in fact, width three) homogeneous ABPs implies a polynomial time reconstruction algorithm for arithmetic formulas due to the reduction from formulas to width three ABPs in [BC92], and this in turn would give a super-polynomial lower bound for formulas (by [FK09]). In hindsight it is no wonder that reconstruction has been accomplished for very restricted circuit models and next we state such models for which efficient reconstruction is known. Then we argue the case for average-case reconstruction that is reconstruction under some distributional assumption on the polyno-
mials computed by a model which is a natural and a promising way to make progress on reconstruction for one of the above mentioned models that lack worst-case reconstruction.

**Previous work on reconstruction**

**Low depth circuits:** A polynomial time reconstruction algorithm for depth two circuits follows from the sparse polynomial interpolation algorithm in [KS01, BT88]. By analysing the rank of the partial derivative matrix, [KS03] gave a randomized reconstruction algorithm for $\Sigma\Pi\Sigma$ circuits, where fan-in of every product gate is bounded by $d$ in time polynomial in the size of the circuit and $2^d$. Prior to this, a polynomial time randomized reconstruction algorithm for set-multilinear depth three circuits followed from [BBB+00]. In both [KS03] and [BBB+00] the output hypothesis is a ROABP. For $\Sigma\Pi\Sigma$ circuits with two product gates [Shp07] gave a randomized reconstruction algorithm over a finite field $F$ that has running time quasi-polynomial in $m, d$ and $|F|$. The running time in [Shp07] is polynomial in $m, |F|$ if the depth three circuit is additionally multilinear. This algorithm was derandomized and extended to depth three circuits with constant number of product gates in [KS09a]. The output hypothesis in [Shp07] is a depth three circuit with two product gates (unless the circuit has a low simple rank$^1$), but it works only over finite fields. Recently, [Sin16] gave a polynomial time randomized reconstruction algorithm for depth three circuits with two product gates over rationals$^2$; the output of the algorithm in [Sin16] is also a depth three circuit with two product gates (unless the simple rank of the circuit is less than a fixed constant). For multilinear depth four circuits with two top level product gates, [GKL12] gave a randomized polynomial time reconstruction algorithm that works over both finite fields and rationals. Recently, for multilinear depth four circuits with $k$ top level product gates [BSV19] gave a deterministic quasi-polynomial time reconstruction algorithm over finite fields, for any constant $k$. If the input is an $n$ variate degree $d$ polynomial $f$ computed by a size $s$ multilinear depth four circuit with $k$ top level product gates then the algorithm in [BSV19] runs in time $2^{(\log nsd(|F|))^{O(1)}}$ and outputs a multilinear depth four circuit of size $2^{(\log s)^{O(1)}}$ computing $f$, for any constant $k$.

**Restricted formulas and ABP:** [MV18] gave a polynomial time reconstruction algorithm for read-once formulas by strengthening the analysis in [SV09], the latter has a quasi-polynomial time reconstruction algorithm for the same model. [FS13] gave a quasi-polynomial time reconstruction algorithms for ROABP, set-multilinear ABP and non-commutative ABP by deran-

---

$^1$The dimension of the span of the linear forms in the two gates after removing their gcd.

$^2$The result holds over characteristic zero fields. We state it for rationals as bit complexity concerns us.
DOMIZING THE ALGORITHM IN [KS03]. PRIOR TO THIS, THE CASE OF NON-COMMUTATIVE ABP RECONSTRUCTION WAS SOLVED IN [AMS08] ASSUMING BLACKBOX ACCESS TO THE INTERNAL GATES OF THE INPUT ABP.

**AVERAGE-CASE RECONSTRUCTION**

AS ALLUDED TO BEFORE, WE KNOW STRONG LOWER BOUNDS THAT ARE ALGEBRAICALLY NATURAL FOR CIRCUIT MODELS LIKE MULTILINEAR FORMULAS, HOMOGENEOUS DEPTH THREE CIRCUITS, HOMOGENEOUS DEPTH FOUR CIRCUITS, REGULAR FORMULAS ETC. FOR ANY ONE OF THESE MODELS LACKING EFFICIENT WORST-CASE RECONSTRUCTION, WE CAN ATTEMPT TO MAKE PROGRESS BY ASKING A SLIGHTLY WEAKER QUESTION: CAN WE DO EFFICIENT RECONSTRUCTION FOR *ALMOST ALL* POLYNOMIALS COMPUTED BY THE MODEL? THIS AMOUNTS TO STUDYING THE RECONSTRUCTION PROBLEM UNDER SOME DISTRIBUTIONAL ASSUMPTIONS ON THE POLYNOMIALS COMPUTED BY THE MODEL. SUCH TYPES OF RECONSTRUCTION ARE CALLED *AVERAGE-CASE RECONSTRUCTION*. SIMILAR AVERAGE-CASE RELAXATIONS OF LEARNING PROBLEMS HAVE BEEN STUDIED IN THE BOOLEAN SETTING, PARTICULARLY FOR DNFs [LSW06, JLSW08]. OFTEN THAN NOT, AN AVERAGE-CASE ALGORITHM IN FACT GIVES A WORST-CASE ALGORITHM FOR INPUTS SATISFYING SOME NATURAL/EASY-TO-STATE NON-DEGENERACY CONDITION WHICH IS ALMOST assuredLY SATISFIED BY A RANDOM INPUT CHOSEN ACCORDING TO ANY REASONABLE DISTRIBUTION. WE FEEL THAT IT IS WORTH KNOWING THESE NON-DEGENERACY CONDITIONS THAT MAKE WORST-CASE RECONSTRUCTION TRACTABLE FOR SOME OF THE MODELS MENTIONED ABOVE.

The distribution under which the reconstruction problem is studied for the above mentioned model should ideally be polynomial time samplable (P-samplable) and interesting from the context of lower bounds in the following sense. A lower bound proof for a model $\mathcal{C}$ shows that an explicit $n$ variate degree $d$ polynomial run is not computable by size $s$ circuits from $\mathcal{C}$, for some $s > \max(n,d)$. Such a proof demonstrates a weakness of the set of $n$ variate degree $d$ polynomials computable by size $s$ circuits from $\mathcal{C}$. In order to exploit the weakness of this set in an average-case reconstruction problem for $\mathcal{C}$, we should ideally define an input distribution that is supported on $n$ variate degree $d$ polynomials computable by size $s$ circuits in $\mathcal{C}$, where $s > \max(n,d)$. For many circuit classes, defining such a distribution is a bit tricky as some of the natural P-samplable distributions tend to be primarily supported on $n$ variate, degree $d$ polynomials where $d$ or $n$ is closely attached to the size $s$ of the circuits produced by

1Typically, the explicit polynomial has degree $d \leq n$, say $d = \sqrt{n}$ or $d = \Theta(n)$ (as in determinant/permanent [Raz09, RY09, GKKS14] or the Nisan-Wigderson design polynomial [KSS14] or the elementary/power symmetric polynomials [NW97, SW01, Kum17] or a variant of the design polynomial [KST16a]).
these distributions, thereby restricting $s$ from being much larger than $\max(n, d)$. For example the distributions defined on multilinear formulas and arithmetic formulas in [GKL11] and [GKQ13] respectively, which we elaborate upon below.

In [GKL11] a randomized polynomial time average-case reconstruction algorithm was given for multilinear formulas picked from a natural distribution: every sum gate computes a random linear combination of the polynomials computed by its two children (sub-formulas), and at every product gate the set of variables is partitioned randomly into two equal size sets between its two children (sub-formulas); the sub-formulas are then constructed recursively. In [GKQ13], a randomized polynomial time reconstruction algorithm was given for fan-in two regular formulas of size $s$ picked from a distribution where the coefficients of the variables in linear polynomials computed at the leaves are chosen independently and uniformly at random from a large subset of $F$. The result in [GKQ13] is an average-case reconstruction algorithm for size $s$ multilinear formulas computing $n$ variate degree $d$ polynomials, where $s = \Theta(n)$. Similarly the result in [GKQ13] is an average-case reconstruction algorithm for size $s$ fan-in two regular formulas computing $n$ variate degree $d$ polynomials, where $s = \Theta(nd^2)$.

In comparison, $n^{\Omega(\log d)}$ size lower bounds are known for multilinear formulas [Raz09] and regular formulas [KSS14].

However, for some classes, like homogeneous $\Sigma\Pi\Sigma$ circuits and homogeneous ABPs these requirements from an input distribution i.e. allowing $s \gg \max(n, d)$ can be mitigated easily, and for both these models there is a simple reduction from reconstruction for homogeneous $\Sigma\Pi\Sigma$ circuits or homogeneous ABPs to reconstruction for (non-homogeneous) $\Sigma\Pi\Sigma$ circuits or ABPs (see Section 6.1.1.2). [KS19] gave a randomized polynomial time reconstruction algorithm for $n$ variate degree $d$ polynomials computed by homogeneous $\Sigma\Pi\Sigma$ circuits satisfying certain non-degeneracy conditions when $n \geq 3d^2$ and the top fan-in is at most $(\frac{n}{3d})^{d^2}$. The result in [KS19] can also be viewed as an average-case reconstruction algorithm for homogeneous depth three circuits sampled from the following distribution: the coefficients of the linear forms computed by the sum gates at the third layer are chosen independently and uniformly at random from a large enough subset of $F$. In this work we study average-case reconstruction of ABPs under a natural distribution (see Definition 1.1). This distribution for average-case ABP reconstruction (Problem 1.2) is quite appropriate to study as it produces $n$ variate degree $d$ polynomials computable by ABPs of size $s \approx wd$ that can potentially be much larger than $\max(n, d)$ with growing width $w$. Recall a $w \geq \frac{n}{2}$ lower bound on homogeneous ABPs computing the power symmetric polynomial $\sum_{i=1}^{n} x_i^d$ is known [Kum17].
Thus the homogeneous ABP reconstruction problem is interesting for \( w < \frac{n}{2} \). However, as mentioned before, in the absence of a super-polynomial lower bound for formulas we cannot hope to do an efficient worst-case reconstruction for homogeneous ABP of even constant width [BC92, FK09]. But, can we do average-case reconstruction for \( w < \frac{n}{2} \)? We make progress in this direction by giving a \((d^w n^3)^{O(1)}\) time average-case reconstruction algorithm for ABPs, where \( \beta \) is the bit length of the coefficients of the input polynomial\(^1\), when \( w \leq \frac{\sqrt{n}}{2} \) (see Theorem 1.3). The time-complexity of our algorithm is better than the brute force algorithm to reconstruct ABPs, when \( w \leq \frac{\sqrt{n}}{2} \) (see the third remark after Theorem 1.3). Also note that ours is a polynomial time average-case reconstruction algorithm for constant width ABPs, and if we can achieve the same complexity for worst-case reconstruction (instead of average-case) then that would imply a super-polynomial lower bound for arithmetic formulas. Moreover our algorithm can also be seen as giving a worst-case reconstruction algorithm for ABPs satisfying certain non-degeneracy conditions (see remarks after Theorems 1.2 and 1.3).

1.1.4 Equivalence testing

**Problem definition and hardness:** Two \( n \) variate polynomials \( f \) and \( g \) are equivalent if there is an invertible \( A \in \mathbb{F}^{n \times n} \) such that \( f = g(A \cdot x) \). Equivalently in the language of group actions we say that \( f \) is equivalent to \( g \) if \( f \) is in the orbit of \( g \) under the action of \( \text{GL}(n, \mathbb{F}) \). Equivalence testing is a natural algorithmic problem in computational algebra where two polynomials \( f \) and \( g \) are given as input\(^2\), and the task is to determine whether \( f \) is equivalent to \( g \). Further, if \( f \) is equivalent to \( g \) then output an invertible \( A \in \mathbb{F}^{n \times n} \) such that \( f = g(A \cdot x) \). The equivalence testing problem is at least as hard as the graph isomorphism problem, even when \( f \) and \( g \) are cubic forms given in dense representation [AS06]. In fact in [AS05, AS06], it is shown that the graph isomorphism problem is polynomial time reducible to commutative \( \mathbb{F} \)-algebra isomorphism problem, which they showed is polynomial time equivalent to cubic form equivalence testing when \( 3 \nmid (|\mathbb{F}| - 1) \) \(^3\). There is a cryptographic application [Pat96] based on the average-case hardness of the equivalence testing problem for bounded degree \( f \) and \( g \) given in dense representation. Also, over finite fields checking whether polynomials \( f \) and \( g \) are equivalent when \( f \) and \( g \) are given as a list of coefficients is in \( \text{NP} \cap \text{coAM} \) [Thi98, Sax06].

\(^1\)Over \( \mathbb{Q} \) our output ABP is over a degree \( w \) extension of \( \mathbb{Q} \), see the third remark after Theorem 1.2.
\(^2\)either as blackbox or in the dense representation as a list of coefficients
\(^3\)The reduction from commutative \( \mathbb{F} \)-algebra isomorphism problem to cubic form equivalence testing works over any field.
**Motivation for equivalence testing of special polynomial families:** Valiant’s extended hypothesis states that the permanent is not a quasi-polynomial (qp)-projection of the determinant. Proving this conjecture is sufficient to show $\text{VP} \neq \text{VNP}$ and is famously known as permanent versus the determinant problem. Further, even showing that the permanent is not a $p$-projection of either the determinant or the IMM polynomial is equivalent to proving $\text{VBP} \neq \text{VNP}$. Geometric complexity theory (GCT) proposed by [MS01] is an approach to resolve this conjecture by using techniques from algebraic geometry, group theory and representation theory. In particular, GCT gives an approach to show that the padded permanent is not in the orbit closure of polynomial size determinant/IMM. Taking inspiration from GCT it is natural to ask whether there is an algorithm to check if a given polynomial is in the orbit closure of polynomial size determinant/IMM. An algorithm for the equivalence testing problem where $f$ is given as a blackbox and $g$ comes from a fixed polynomial family like the determinant or the IMM is a natural step towards the search for such an algorithm.

**Previous work**

In [Kay11] the equivalence testing problem was studied when $f$ is given as a blackbox and $g$ comes from a fixed polynomial family, and polynomial time randomized algorithms were given when $g$ is a power symmetric polynomial, an elementary symmetric polynomial or a sum of products polynomials. Later in [Kay12a] polynomial time randomized equivalence testing algorithms were given for the permanent and the determinant polynomials using the Lie algebras (Definition 2.17) of these polynomials. For all polynomials except the determinant the algorithms in [Kay11, Kay12a] work over all fields namely $\mathbb{C}$, $\mathbb{Q}$, and finite fields of large characteristic, but for the determinant polynomial the equivalence testing algorithm in [Kay12a] only works over $\mathbb{C}$. All these polynomials considered for equivalence testing are well-studied from the context of arithmetic circuit lower bounds and have been used to prove lower bounds for different circuit models. In this work we give a polynomial time randomized equivalence testing algorithm for IMM (Theorem 1.1b) using the Lie algebra of IMM. Our approach in using the Lie algebra of IMM differs from that of [Kay12a] and a comparison between the two techniques is given in Chapter 4 where we present our algorithm for equivalence testing of IMM. We also give a polynomial time randomized equivalence testing algorithm for determinant over finite fields but the output matrix is over a small extension of the base field (see Theorem 6.1). This was recently subsumed by the work of [GGKS19] where they gave a polynomial time randomized equivalence testing algorithm for determinant over
finite fields\footnote{with the following condition on the characteristic of } and $\mathbb{Q}$ by reducing it to a well-studied problem in symbolic computation which they call the ‘full matrix algebra isomorphism’ problem \cite{Ron87, Ron90, BR90, IRS12}. In the case of finite fields their output is over the base field while in the case of $\mathbb{Q}$ the output is over a small extension of $\mathbb{Q}$. Polynomial time randomized algorithms for block-diagonal permutation-scaling equivalence testing\footnote{The transformation matrix $A$ is the product of a block-diagonal permutation matrix and an invertible scaling matrix.} of another important polynomial family, namely the Nisan-Wigderson polynomial family is given in \cite{GS19}.

### 1.2 Our contributions

Having introduced four major problems in arithmetic circuit complexity, in this section we motivate and state our contributions towards each of these problems. From here on, we refer to a linear polynomial as an affine form and a linear polynomial with a zero constant term as a linear form.

#### 1.2.1 Equivalence test for IMM

Our main result is a polynomial time randomized algorithm for equivalence testing of $\text{IMM}_{w,d}$ (see Section 2.3). Theorem 1.1a gives an algorithm to check whether $f$ is an affine projection of the polynomial $\text{IMM}_{w,d}$ via a full rank transformation (see Definition 2.14). Using known results on variable reduction and translation equivalence test (see Section 2.7) proving Theorem 1.1a reduces in randomized polynomial time to giving an equivalence test for $\text{IMM}_{w,d}$ which we state in Theorem 1.1b.

**Theorem 1.1a** Given blackbox access to an $m$ variate polynomial $f \in \mathbb{F}[x]$ of degree $d \in [5, m]$, the problem of checking if there exist a $w \in \mathbb{N}^{d-1}$, a $B \in \mathbb{F}^{n \times m}$ of rank $n$ equal to the number of variables in $\text{IMM}_{w,d}$, and a $b \in \mathbb{F}^n$ such that $f = \text{IMM}_{w,d}(Bx + b)$\footnote{A variable set $x = \{x_1, \ldots, x_m\}$ is treated as a column vector $(x_1 \ldots x_m)^T$ in the expression $Bx + b$. The affine form entries of the column $Bx + b$ are then plugged in place of the variables of $\text{IMM}_{w,d}$ (following a variable ordering, like the one mentioned in Section 2.3).}, can be solved in randomized $(m\beta)^O(1)$ time where $\beta$ is the bit length of the coefficients of $f$. Further, with probability at least $\frac{1}{n}$, the following is true: the algorithm returns a $w \in \mathbb{N}^{d-1}$, a $B \in \mathbb{F}^{n \times m}$ of rank $n$, and a $b \in \mathbb{F}^n$ such that $f = \text{IMM}_{w,d}(Bx + b)$ if such $w$, $B$ and $b$ exist, else it outputs ‘no such $w$, $B$ and $b$ exist’.

**Theorem 1.1b (Equivalence test for IMM)** Given blackbox access to a homogeneous $n$ variate polynomial $f \in \mathbb{F}[x]$ of degree $d \in [5, n]$, where $|x| = n$, the problem of checking if there exist a
\( w \in \mathbb{N}^{d-1} \) and an invertible \( A \in \mathbb{F}^{n \times n} \) such that \( f = \text{IMM}_{w,d}(Ax) \), can be solved in randomized \((n\beta)^{O(1)}\) time where \( \beta \) is the bit length of the coefficients of \( f \). Further, with probability at least \( 1 - n^{-\Omega(1)} \) the following holds: the algorithm returns a \( w \in \mathbb{N}^{d-1} \), and an invertible \( A \in \mathbb{F}^{n \times n} \) such that \( f = \text{IMM}_{w,d}(Ax) \) if such \( w \) and \( A \) exist, else it outputs ‘no such \( w \) and \( A \) exist’.

Remarks: The algorithms in Theorems 1.1a and 1.1b work over \( \mathbb{Q} \) and finite fields of large enough characteristic, and is obtained by analysing the irreducible invariant subspaces of the Lie algebra of a polynomial \( f \) equivalent to \( \text{IMM}_{w,d} \) (see Definition 2.17) which are intimately related to the layer spaces of a full-rank ABP (see Definition 2.5) computing \( f \). Following are a few additional remarks on Theorem 1.1b. Suppose \( f = \text{IMM}_{w,d}(Ax) \), where \( A \) is an invertible matrix in \( \mathbb{F}^{n \times n} \) and \( w = (w_1, w_2, \ldots, w_{d-1}) \).

1. Irreducibility of \( \text{IMM}_{w,d} \): We can assume without loss of generality that \( w_k > 1 \) for every \( k \in [d-1] \), implying \( \text{IMM}_{w,d} \) is an irreducible polynomial. If \( w_k = 1 \) for some \( k \in [d-1] \) then \( \text{IMM}_{w,d} \) is reducible, in which case we use the factorization algorithm in [KT90] to get blackbox access to the irreducible factors of \( f \) and then apply Theorem 1.1b to each of these irreducible factors (Section 4.1.1 has more details on this).

2. Uniqueness of \( w \) and \( A \): Assuming \( w_k > 1 \) for every \( k \in [d-1] \), it would follow from the proof of the theorem that \( w \) is unique in the following sense: if \( f = \text{IMM}_{w',d}(A'x) \), where \( A' \in \mathbb{F}^{n \times n} \) is invertible, then either \( w' = w \) or \( w' = (w_{d-1}, w_{d-2}, \ldots, w_1) \). The invertible transformation \( A \) is also unique up to the group of symmetries (see Definition 2.16) of \( \text{IMM}_{w,d} \): if \( \text{IMM}_{w,d}(Ax) = \text{IMM}_{w,d}(A'x) \) then \( AA'^{-1} \) is in the group of symmetries of \( \text{IMM}_{w,d} \). In Chapter 5, we determine this group and show that \( \text{IMM}_{w,d} \) is characterized by it.

3. No knowledge of \( w \): The algorithm does not need prior knowledge of the width vector \( w \), it only knows the number of variables \( n \) and the degree \( d \) of \( f \). The algorithm is able to derive \( w \) from blackbox access to \( f \).

4. A related result in [Mur19, Gro12]: Another useful definition of the iterated matrix multiplication polynomial is the trace of a product of \( d \), \( w \times w \) symbolic matrices – let us denote this polynomial by \( \text{IMM}'_{w,d} \). Both the variants, \( \text{IMM}'_{w,d} \) and \( \text{IMM}_{w,d} \), are well-studied in the literature and their circuit complexities are polynomially related. However, an equivalence test for one does not immediately give an equivalence test for the other. This is partly because the group of symmetries of \( \text{IMM}'_{w,d} \) and \( \text{IMM}_{w,d} \) are not equal (see Chapter 5 for a comparison).
Let $x_1, \ldots, x_d$ be the sets of variables in the $d$ matrices of $\text{IMM}'_{w,d}$ respectively. A polynomial $f(x_1, \ldots, x_d)$ is said to be \textit{multilinearly equivalent} to $\text{IMM}'_{w,d}$ if there are invertible $w \times w$ matrices $A_1, \ldots, A_d$ such that $f = \text{IMM}'_{w,d}(A_1x_1, \ldots, A_dx_d)$. Grochow [Gro12] showed the following result: Given the knowledge of the variable sets $x_1, \ldots, x_d$, an oracle to find roots of univariate polynomials over $\mathbb{C}$ and blackbox access to a polynomial $f$, there is a randomized algorithm to check whether $f$ is multilinearly equivalent to $\text{IMM}'_{w,d}$ using $O(1)$ operations over $\mathbb{C}$. Recently, using the techniques from our work, over $\mathbb{Q}$ and finite fields of large enough characteristic, [Mur19] reduced the equivalence testing of $\text{IMM}'_{w,d}$ to the same problem considered by [Gro12] that is multilinear equivalence testing of $\text{IMM}'_{w,d}$. Further using the determinant equivalence testing algorithm in [GGKS19], [Mur19] is able to solve this over $\mathbb{Q}$ and any finite field of large enough characteristic. Over $\mathbb{Q}$ the output matrices are over a degree $w$ extension of $\mathbb{Q}$, and over finite fields the output matrices are over the base field.

1.2.2 Average-case linear matrix factorization and low width ABP circuit reconstruction

Choice of Distribution and Problems

We study average-case version of two related problems \textit{linear matrix factorization} (LMF) and ABP reconstruction. The average-case LMF problem aids us in making progress on the average-case ABP reconstruction problem. A matrix is called a \textit{linear matrix} if all its entries are affine forms and an ABP of width $w$ and length $d$ can be modelled as a product of linear matrices $X_1 \cdot X_2 \cdots X_d$, where $X_1, X_d$ are row and column linear matrices of size $w$ and $X_2, \ldots, X_{d-1}$ are $w \times w$ linear matrices. We denote an ABP of width $w$ and length $d$ computing an $n$ variate polynomial as a $(w, d, n)$-ABP. Similarly a $(w, d, n)$-matrix product is a product of $d$, $w \times w$ linear matrices with affine forms in $n$ variables as entries. Notice that in the case of a $(w, d, n)$-matrix product, $X_1$ and $X_d$ are $w \times w$ matrices and not row and column matrices as in the case of an ABP. A simpler question than reconstruction of ABPs is the LMF problem, where blackbox access to $w^2$ correlated entries of a $(w, d, n)$-matrix product $F$ is given as input and the outputs are $d', w \times w$ linear matrices $Y_1, \ldots, Y_{d'}$ satisfying $F = Y_1 \cdots Y_{d'}$.

In order to study average-case complexity of these problems, we define a distribution on polynomials computed by ABPs or matrix products in Definition 1.1. The distributions on ABPs and matrix products are posed over $\mathbb{F}_q$ – a finite field, and it can be defined over $\mathbb{Q}$ by replacing $\mathbb{F}_q$ in Definition 1.1 with an appropriate choice of $S \subseteq \mathbb{Q}$. In Definition 1.1, a
random linear matrix over $\mathbb{F}_q$ is a linear matrix, where the coefficients of affine forms are chosen independently and uniformly at random from $\mathbb{F}_q$.

**Definition 1.1 (Random ABP and random matrix product)** Given the parameters $w, d$ and $n$, a random $(w, d, n)$-ABP over $\mathbb{F}_q$ is a $(w, d, n)$-ABP $X_1 \cdots X_d$ where $X_1, \ldots, X_d$ are random linear matrices over $\mathbb{F}_q$. Similarly a random $(w, d, n)$-matrix product over $\mathbb{F}_q$ is a $(w, d, n)$-matrix product $X_1 \cdots X_d$ where $X_1, \ldots, X_d$ are random linear matrices over $\mathbb{F}_q$.

Note that there is a sampling algorithm which when given the parameters $w, d, n$ outputs a random $(w, d, n)$-ABP or a random $(w, d, n)$-matrix product over $\mathbb{F}_q$, in time $(w d n \log q)^O(1)$.

Average-case LMF and average-case ABP reconstruction problem can now be posed as follows.

**Problem 1.1 (Average-case LMF)** Design an algorithm which when given blackbox access to $w^2$, $n$ variate degree $d$ polynomials $\{f_{st}\}_{s,t \in [w]}$ that constitute the entries of a random $(w, d, n)$-matrix product $F$ over $\mathbb{F}_q$, outputs $d, w \times w$ linear matrices $Y_1, \ldots, Y_d$ over $\mathbb{F}_q$ (or a small extension of $\mathbb{F}_q$) such that $F = Y_1 \cdot Y_2 \cdots Y_d$ with high probability. The desired running time of the algorithm is $(w d n \log q)^O(1)$.

**Problem 1.2 (Average-case ABP reconstruction)** Design an algorithm which when given blackbox access to an $n$ variate degree $d$ polynomial $f$ computed by a random $(w, d, n)$-ABP over $\mathbb{F}_q$, outputs a $(w, d, n)$-ABP over $\mathbb{F}_q$ (or a small extension of $\mathbb{F}_q$) computing $f$ with high probability. The desired running time of the algorithm is $(w d n \log q)^O(1)$.

The probability of an algorithm that solves Problem 1.1 (respectively Problem 1.2) is taken over the random choice of the input random $(w, d, n)$-matrix product (respectively random $(w, d, n)$-ABP) as well as over the random bits used by the reconstruction algorithm, if it is randomized.

**Our Results**

In Theorems 1.2 and 1.3, $\text{char}(\mathbb{F}_q) \geq (dn)^7$ and $\text{char}(\mathbb{F}_q) \nmid w$ (see the first remark after Theorem 1.2), and $\mathbb{L}$ the extension field $\mathbb{F}_{q^w}$. ($\mathbb{L}$ can be constructed from a basis of $\mathbb{F}_q$ using a randomized algorithm running in $(w \log q)^O(1)$ time [vzGG03].) Theorem 1.2 solves Problem 1.1 for $n \geq 2w^2$.

**Theorem 1.2 (Average-case LMF)** For $n \geq 2w^2$, there is a randomized algorithm that takes as input blackbox access to $w^2$, $n$ variate, degree $d$ polynomials $\{f_{st}\}_{s,t \in [w]}$ that constitute the entries of a random $(w, d, n)$-matrix product $F = X_1 \cdot X_2 \cdots X_d$ over $\mathbb{F}_q$, and with probability
$1 - \frac{(wd)^{O(1)}}{q}$ returns $w \times w$ linear matrices $Y_1, Y_2, \ldots, Y_d$ over $\mathbb{L}$ satisfying $F = \prod_{i=1}^{d} Y_i$. The algorithm runs in $(dn \log q)^{O(1)}$ time and queries the blackbox at points in $\mathbb{L}^n$.

Remarks:

- The need for going to an extension field is removed owing to a recent result by [GGKS19]. The algorithm in Theorem 1.2 reduces the average-case LMF problem to equivalence testing for $\text{Det}_w$ for which we give a $(w \log q)^{O(1)}$ time algorithm over $F_q$, which outputs a matrix over $\mathbb{L}$. Recently [GGKS19] gave a polynomial time randomized algorithm for determinant equivalence test over finite fields where the output matrix is over the base field itself instead of an extension field like in our case. Using the equivalence testing algorithm of [GGKS19], the algorithm in Theorem 1.2 outputs a matrix product over $F_q$ and queries the blackbox at points in $\mathbb{F}_q^n$.

- **Average-case LMF over $\mathbb{Q}$:** [GGKS19] also gave an algorithm for equivalence testing of $\text{Det}_w$ over $\mathbb{Q}$ where the output matrix is over a degree $w$ extension of $\mathbb{Q}$. Using the equivalence testing algorithm of [GGKS19], the algorithm in Theorem 1.2 works over $\mathbb{Q}$ where the output is over a degree $w$ extension of $\mathbb{Q}$.

- The constraint on $\text{char}(F_q)$ is a bit arbitrary, the results hold as long as $|F_q|$ and $\text{char}(F_q)$ are sufficiently large polynomial functions in $d$ and $n$. The requirement of a finite field $F_q$ and $\text{char}(F_q) \nmid w$ is needed in the proof of Theorem 6.1 which gives an equivalence test for the determinant polynomial over $F_q$. But owing to a recent result by [GGKS19], the algorithm in Theorem 1.2 also works over $\mathbb{Q}$. We elaborate on this in the next point.

- **Pure matrix product:** A $(w,d,n)$-matrix product $X_1 \cdot X_2 \cdots X_d$ is pure if it satisfies the following properties:

  1. For every $i \in [d]$, $X_i$ is a full-rank linear matrix. A linear matrix is a full-rank linear matrix if the degree one homogeneous parts of its affine form entries are $F_q$-linearly independent. This is a stronger notion than $\det(X_i) \neq 0$ in $F(x)$, where $\det(X_i)$ is the determinant of $X_i$. In particular it is easy to see that if the affine forms in $X_i$ are $F_q$-linearly independent then $\det(X_i) \neq 0$ but vice-versa is not always true.

  2. For every $i, j \in [d]$ and $i \neq j$, $\det(X_i)$ and $\det(X_j)$ are coprime.

---

1For $q \geq (dn)^7$ the probability is $1 - \frac{1}{(dn)^{17}}$. 

22
3. For every \( i, j \in [d] \) and \( i < j \), the \( w^2 \) polynomial entries of the partial product \( X_{i+1} \cdots X_j \) are \( \mathbb{F}_q \)-linearly independent modulo the affine forms in the first row and column of \( X_i \).

It can be shown (see Claim 6.1 and Claim 6.2) that a random \((w, d, n)\)-matrix product is a pure matrix product (in short, a pure product) with high probability, for \( n \geq 2w^2 \). Theorem 1.2 actually gives a polynomial time linear matrix factorization algorithm for a pure product.

- **Uniqueness of factorization:** The proof of the theorem also shows that linear matrix factorization of a pure product is unique in the following sense – there are \( C_i, D_i \in \text{GL}(w, \mathbb{L}) \) such that \( Y_i = C_i \cdot X_i \cdot D_i \) for every \( i \in [d] \). Moreover, there are \( c_1, \ldots, c_{d-1} \in \mathbb{L}^\times \) satisfying \( C_1 = D_d = I_w, \) \( D_i \cdot C_{i+1} = c_i I_w \) for \( i \in [d-1] \), and \( \prod_{i=1}^{d-1} c_i = 1 \). At a very high level, it is this uniqueness feature that guides the algorithm in finding a factorization. Such a factorization need not be unique if only the first two properties are satisfied. For instance\(^2\),

\[
\begin{bmatrix}
  x_1 & x_2 \\
  x_3 & x_4
\end{bmatrix}
\begin{bmatrix}
  2x_3 - x_2 & x_4 \\
  x_1 & x_3
\end{bmatrix}
= \begin{bmatrix}
  x_3 & x_1 \\
  x_4 & 2x_3 - x_2
\end{bmatrix}
\begin{bmatrix}
  x_1 & x_2 \\
  x_3 & x_4
\end{bmatrix}
= \begin{bmatrix}
  2x_1 x_3 & x_1 x_4 + x_2 x_3 \\
  2x_3^2 - x_2 x_3 + x_1 x_4 & 2x_3 x_4
\end{bmatrix}.
\]

Using Theorem 1.2, Theorem 1.3 addresses Problem 1.2 for \( n \geq 4w^2 \) and \( d \geq 5 \). The constraints on the field in Theorem 1.3 are similar to Theorem 1.2; also see the first two remarks above.

**Theorem 1.3 (Average-case ABP reconstruction)** For \( n \geq 4w^2 \) and \( d \geq 5 \), there is a randomized algorithm that takes as input blackbox access to an \( n \) variate, degree \( d \) polynomial \( f \) computed by a random \((w, d, n)\)-ABP over \( \mathbb{F}_q \), and with probability \( 1 - \frac{(wd)^{O(1)}}{q} \) returns a \((w, d, n)\)-ABP over \( \mathbb{L} \) computing \( f \). The algorithm runs in time \((d^{w^3} n \log q)^{O(1)}\) and queries the blackbox at points in \( \mathbb{L}^n \).

**Remarks:**

1. In Theorem 1.3, the need for going to an extension field arises from the use of the algorithm in Theorem 1.2. Hence, using the equivalence testing algorithm for \( \text{Det}_w \) by\(^3\)

---

\(^1\)The choice of the first row and column are arbitrary. The analysis holds if the entries of \( X_{i+1} \cdots X_j \) are \( \mathbb{F}_q \)-linearly independent modulo the affine forms in some row and column of \( X_i \).

\(^2\)We thank Rohit Gurjar for showing us a similar example, but with non-coprime determinants.

\(^3\)For \( q \geq (dn)^7 \) the probability is \( 1 - \frac{1}{(dn)^{w^2}} \).
The algorithm in Theorem 1.3 outputs a \((w,d,n)\)-ABP over \(F_q\) computing \(f\), and queries the blackbox at points in \(F_q^n\) (see the first remark after Theorem 1.2).

2. **Comparison to Theorem 1.1a**: If \(n \geq w^2d\) then with high probability a polynomial computed by a random \((w,d,n)\)-ABP is an affine projection of \(\text{IMM}_{w,d}\) via a full-rank transformation. Hence, the algorithm in Theorem 1.1a also gives an efficient average-case reconstruction algorithm for ABPs when \(w \leq \sqrt{n/d}\). Observe that, under this width constraint, the size \(s \approx wd\) of an ABP is upper bounded by \(\max(n,d)\). Whereas, in Theorem 1.3 we give a reconstruction algorithm for \(w \leq \sqrt{n/2}\) (independent of \(d\)), and hence the size of the ABPs in this case can be \(s = \Theta(\sqrt{n})\). To highlight this improvement, if we set \(d = \Theta(n)\) (as in several lower bound results [Kum17, KST16a, SW01, NW97]) then the width constraint in Theorem 1.1a reduces to \(w = O(1)\) and size becomes \(\Theta(n)\), whereas the size of the ABPs (width up to \(\sqrt{n^2}\)) in Theorem 1.3 is \(\Theta(n^{1.5})\) which is significantly closer to the best known \(\Omega(n^2)\) lower bound for homogeneous ABPs. Also, it is because of the independence of \(d\) on the width constraint that we could infer that the same time complexity for worst-case reconstruction of constant width homogeneous ABP would imply a super-polynomial formula lower bound. This is because the process of homogenizing a non-homogeneous ABP to a homogeneous ABP (described in Section 6.1.1.2) bloats up the degree \(d\). These factors underscore the importance of getting rid of the dependence on \(d\) from the width constraint. On the flip side though, the algorithm in Theorem 1.1a works even when the widths of the intermediate matrices in the ABP vary and runs in time \((wdn \log q)^{O(1)}\), whereas the algorithm in this work cannot handle varying width ABP and has time complexity \((d^{w^3} n \log q)^{O(1)}\).

3. **Comparison to brute force algorithm**: A brute-force algorithm to reconstruct a \((w,d,n)\)-ABP over \(F_q\) takes time \(q^{\Theta(w^2dn)}\). The algorithm in Theorem 1.3 takes time sub-exponential in the quantity \(w^2dn\) when \(w \leq \sqrt{n/d}\). Note that we can interpolate a polynomial computed by a \((w,d,n)\)-ABP over \(F_q\) in \((d^m \log q)^{O(1)}\) time, but knowing the coefficients of the polynomial does not give us any immediate information about the \((w,d,n)\)-ABP that computes it – this point is related to the hardness of the MCSP problem and reconstruction under dense representation of the input polynomial mentioned before. Hence, if we want a \((w,d,n)\)-ABP representation for the input polynomial over \(F_q\) then even a \((d^m \log q)^{O(1)}\) time reconstruction algorithm is non-trivial as \(d^m\) is sub-exponential in \(w^2dn\) for any \(d = n^{\Omega(1)}\). The complexity of our algorithm is \((d^{w^3} n \log q)^{O(1)}\) which is also sub-exponential in \(w^2dn\) for \(w \leq \sqrt{n/2}\). For instance, if \(m = w^2dn\), \(w = \sqrt{n/2}\) and \(d = \Theta(n)\) then the trivial complexity is \(\exp(m)\) and our algorithm’s time complexity is...
exp(\sqrt{m}).

4. **Increase in time-complexity from LMF to ABP reconstruction:** There is one step in the algorithm that finds the affine forms in \(X_1\) and \(X_d\) by solving systems of polynomial equations over \(\mathbb{F}_q\), and this takes \(d^O(w^3)\) field operations. Except this step, every other step runs in \((dn \log q)^{O(1)}\) time. If the complexity of this step is improved then the overall time complexity of the algorithm will also come down.

5. **Not pseudorandom:** Consider a formal \((w, d, n)\)-ABP where the coefficients of the affine forms are distinct \(y\)-variables, and let \(h(x, y)\) be the polynomial computed by this ABP. Here, \(m \equiv |y| = (n + 1) \cdot (w^2(d - 2) + 2w)\). If \(w = O(\sqrt{n})\), the family \(H = \{h(x, b) : b \in \mathbb{F}_q^m\}\) is not pseudorandom under the distribution defined by \(b \in_r \mathbb{F}_q^m\). That is there is a polynomial time randomized algorithm that takes as input the \(\binom{n+d}{d}\) coefficients of an \(n\) variate, degree \(d\) polynomial \(f\) such that either \(f\) is sampled from \(H\) by picking a \(b \in_r \mathbb{F}_q^m\) or \(f\) is a random \(n\) variate, degree \(d\) polynomial\(^1\) and with high probability does the following: outputs 1 if the input polynomial \(f \in H\) and 0 otherwise. The randomized algorithm (Theorem 2.6 and 3.9 in [HW99]) checks if the variety of \(f\) denoted as \(V(f)\) \(^2\) has a large subspace in \((dw^2n \log q)^{O(1)}\) time – if \(f\) is sampled from \(H\) then the \(w\) affine forms in \(X_1\) are linearly independent with high probability and the variety of \(f = h(x, b)\) has a subspace of dimension \(n - w\) over \(\mathbb{F}_q\) whereas the variety of a random polynomial with high probability does not have such a subspace of dimension \(n - w\) over \(\mathbb{F}_q\). Observe that \((dw^2n \log q)^{O(1)} = d^O(n)\) for \(w = O(\sqrt{n})\), and so the algorithm takes time polynomial in the number of monomials in \(f\) to distinguish it from a random \(n\) variate, degree \(d\) polynomial thereby implying that \(H\) is not a pseudorandom family. However, a family not being pseudorandom under a distribution does not say much a priori about average-case reconstruction under the same distribution for the family. The latter is presumably a much harder problem for arbitrary non-pseudorandom polynomial families.

6. **Non-degenerate ABP:** Similar to pure product, we can state a set of non-degeneracy conditions such that the algorithm in Theorem 1.3 (with a slight modification) solves the reconstruction problem for ABPs satisfying these conditions. These non-degeneracy conditions are stated in Section 6.4.3, and the proof of Theorem 1.3 shows that a random \((w, d, n)\)-ABP satisfies them with high probability, for \(n \geq 4w^2\) and \(d \geq 5\).

---

\(^1\)The \(\binom{n+d}{d}\) coefficients of a random \(n\) variate, degree \(d\) polynomial are chosen independently and uniformly at random from \(\mathbb{F}_q\).

\(^2\)The variety of an \(n\) variate polynomial \(f(x)\) is defined as, \(V(f) \equiv \{a \in \mathbb{F}_q^n \mid f(a) = 0\}\).
1.2.3 Lower bounds for IMM

Separation between multilinear depth three circuits and ROABPs

Our main result is an exponential lower bound on the size of multilinear depth three formulas computing $\text{IMM}_{w,d}$.

**Theorem 1.4** Any multilinear depth three circuit (over any field) computing $\text{IMM}_{w,d}$ has top fan-in $w^{\Omega(d)}$ for $w \geq 6$.

Theorem 1.4 is proved by introducing a novel variant of the partial derivatives measure called skewed partial derivatives (see Definition 2.13) that is inspired by [Nis91, Raz09] and [NW97]. Theorem 1.4 also implies a lower bound for determinant, see Corollary 7.1. As a consequence of the proof of Theorem 1.4 we also get an exponential separation between multilinear depth three and multilinear depth four circuits.

**Theorem 1.5** There is an explicit family of $O(w^2d)$-variate polynomials of degree $d$, $\{f_d\}_{d \geq 1}$, such that $f_d$ is computable by a $O(w^2d)$-sized multilinear depth four circuit with top fan-in one (i.e. a $\Pi_2 \Sigma_2 \Pi$ circuit) and every multilinear depth three circuit computing $f_d$ has top fan-in $w^{\Omega(d)}$ for $w \geq 11$.

In our Masters thesis and also as part of [KNS16] we proved an exponential size lower bound on ROABPs computing a polynomial that is computed by a small multilinear depth three circuit. We state this result in Theorem 1.6 but do not include its proof in this thesis. In Theorem 1.6, $\mathbb{P}$ denotes the set of primes.

**Theorem 1.6** There is an explicit family of $3n$-variate polynomials $\{f_n\}_{n \in \mathbb{P}}$ over any field $\mathbb{F}$ such that the following holds: $f_n$ is computable by a multilinear depth three circuit $C$ over $\mathbb{F}$ with top fan-in two and any ROABP over $\mathbb{F}$ computing $f_n$ has width $2^{\Omega(n)}$.

The explicit polynomial in Theorem 1.6 is constructed from expander graphs, and its expansion property is used to show that the evaluation dimension (see Definition 2.12) of this polynomial with respect to any subset of size $\frac{n}{10}$ is at least $2^{\Omega(n)}$. Whereas for every ROABP there is a set $S$ of size $\frac{n}{10}$ such that the evaluation dimension of the polynomial computed by the ROABP is at most its width. Since $\text{IMM}_{w,d}$ can be computed by a $(wd)^{O(1)}$ size ROABP, Theorems 1.4 and 1.6 together imply a complete separation between polynomial size multilinear depth three circuits and ROABPs.
Lower bounds on interval set-multilinear formula

Another interesting restriction in arithmetic circuit computation is non-commutativity. The variables do not commute under multiplication in a non-commutative polynomial, for example \( x_1x_2 \) and \( x_2x_1 \) are distinct non-commutative monomials. [Nis91] used the rank of the partial derivative matrix to prove exponential size lower bound on non-commutative ABPs computing a polynomial computed by linear size non-commutative circuits. The result by [Nis91] shows an exponential separation between circuits and ABPs in the non-commutative setting. We focus on separating the computational powers of ABPs and homogeneous formulas for non-commutative computation, which is an important open problem. It turns out that to show a separation between homogeneous non-commutative formulas and non-commutative ABPs it suffices to prove a super-polynomial lower bound on the size of a more restricted multilinear formula computing \( \text{IMM}_{w,d} \) (see Section 7.3.1). We call them interval set-multilinear formulas and they are defined as follows.

**Definition 1.2 (Interval set-multilinear formula)** An interval set-multilinear formula \( \phi \) in variables \( x = \biguplus_{i \in [d]} x_i \) computes a set-multilinear polynomial in variable sets \( x_1, \ldots, x_d \). A node \( v \) in \( \phi \) is associated with an interval \( I \subseteq [d] \) such that \( v \) computes a set-multilinear polynomial in the variable sets \( \{x_i, i \in I\} \). Let \( I = \{i_1, i_2\} \) where \( 1 \leq i_1 \leq i_2 \leq d \). If \( v \) is product node with children \( v_1 \) and \( v_2 \) then there is an \( i_3 \in [i_1, i_2 - 1] \) such that the intervals associated with \( v_1 \) and \( v_2 \) are \( [i_1, i_3] \) and \( [i_3 + 1, i_2] \) respectively. If \( v \) is an addition node with children \( v_1 \) and \( v_2 \) then the same interval \( I \) as its parent is associated with \( v_1 \) and \( v_2 \).

Note that in the above definition we have assumed the fan-in of the gates is equal to two. This is without loss of generality and the model can be defined appropriately for arbitrary arity, but for our results we will stick to fan-in two. An example of an interval set-multilinear formula is the multilinear formula of size \( wO(\log d) \) computing \( \text{IMM}_{w,d} \) obtained via the divide and conquer approach. We consider a natural restriction of interval set-multilinear formulas as defined below. In Definition 1.3, if \( I = [i_1, i_2] \), where \( 1 \leq i_1 \leq i_2 \leq d \), then \( |I| = i_2 - i_1 + 1 \).

**Definition 1.3 (\( \alpha \)-balanced interval set-multilinear formula)** Let \( \alpha \in (0, \frac{1}{2}) \). Then an \( \alpha \)-balanced interval set-multilinear formula satisfies the following: if \( I \) is the interval associated with the product gate \( v \) and \( I_1, I_2 \) are the intervals associated with the children of \( v \) then the lengths of \( I_1 \) and \( I_2 \) (denoted \( |I_1| \) and \( |I_2| \) respectively) are at least \( \lceil \alpha |I| \rceil \).

We prove the following size lower bound on \( \alpha \)-balanced interval set-multilinear formulas computing \( \text{IMM}_{w,d} \).

\(^1\)Each monomial in a set-multilinear polynomial in variable sets \( x_1, \ldots, x_d \) has exactly one variable from every set \( x_i, i \in [d] \).
Theorem 1.7 The size of any $\alpha$-balanced interval set-multilinear formula computing $\text{IMM}_{w,d}$ is $w^{\Omega\left(\frac{\log d}{\log \alpha}\right)}$.

Observe that the lower bound in Theorem 1.7 is super-polynomial for $\alpha = d^{\frac{1}{\omega(1)}}$, and $\alpha < \frac{1}{2}$.

1.2.4 PIT for superposition of set-multilinear depth three circuits

The work stated in this section is in our masters thesis and is also part of [KNS16], and we do not include the proof of Theorem 1.8 (stated below) in this thesis. With a view to making progress on PIT for multilinear depth three circuits we proposed an intermediate model between set-multilinear depth three and multilinear depth three circuits.

Definition 1.4 (Superposition of set-multilinear depth three circuits) A multilinear depth three circuit $C$ over a field $F$ is a superposition of $t$ set-multilinear depth three circuits over variables $x = \bigcup_{i=1}^{t} y_i$, if for every $i \in [t]$, $C$ is a set-multilinear depth three circuit in $y_i$ variables over the field $F(x \setminus y_i)$. The sets $y_1, ..., y_t$ are called the base sets of $C$.

In Theorem 1.6 we show that the hard polynomial family $\{f_n\}_{n \in \mathbb{N}}$ is computed by a multilinear depth three circuit $C$ over $F$ with top fan-in two and simultaneously $C$ is also a superposition of three set-multilinear depth three circuits. Additionally as part of Theorem 1.6 in [KNS16] we also show that there is an explicit family of $3n$ variate polynomials $\{g_n\}_{n \in F}$ over any field $F$ such that $g_n$ is computable by a multilinear depth three circuit $C$ over $F$ with top fan-in three and simultaneously $C$ is also a superposition of two set-multilinear depth three circuits whereas any ROABP over $F$ computing $g_n$ has width $2^{\Omega(n)}$.

The hard polynomial families in Theorem 1.6 are efficiently computed by a special type of multilinear depth three circuits - they are both superpositions of constantly many set-multilinear depth three circuits and simultaneously a sum of constantly many set-multilinear depth three circuits. Here is an example of a circuit from this class.

$$C(x, y) = (1 + 3x_1 + 5y_2)(4 + x_2 + y_1) + (9 + 6x_1 + 4y_2)(3 + 2x_2 + 5y_1)$$

$$+ (6 + 9x_1 + 4y_1)(2 + 5x_2 + 3y_2) + (3 + 6x_1 + 9y_1)(5 + 8x_2 + 2y_2).$$

$C(x, y)$ is a superposition of two set-multilinear depth three circuits with base sets $x = \{x_1\} \cup \{x_2\}$ and $y = \{y_1\} \cup \{y_2\}$. But $C(x, y)$ is also a sum of two set-multilinear depth three circuits with $\{x_1, y_2\}$, $\{x_2, y_1\}$ being the partition in the first set-multilinear depth three circuit (corresponding to the first two products) and $\{x_1, y_1\}$, $\{x_2, y_2\}$ the partition in the second set-multilinear depth three circuit (corresponding to the last two products). For such a subclass of
multilinear depth three circuits, in our MSc.(Engineering) thesis we gave a quasi-polynomial
time algorithm for blackbox PIT.

**Theorem 1.8** Let $C_{n,m,l,s}$ be a subset of multilinear depth three circuits computing $n$ variate
polynomials such that every circuit in $C_{n,m,l,s}$ is a superposition of at most $m$ set-multilinear
depth three circuits and simultaneously a sum of at most $l$ set-multilinear depth three circuits,
and has top fan-in bounded by $s$. Then there is a blackbox PIT algorithm for $C_{n,m,l,s}$ running in
$(ns)^{O(lm \log s)}$ time.

When $m$ and $l$ are bounded by $(\log ns)^{O(1)}$, we get a quasi-polynomial time algorithm for
blackbox PIT. The algorithm in Theorem 1.8 is obtained by extending the shift and rank
concentration technique of [ASS13].

### 1.3 Organization

In Chapter 2, we formally define the different arithmetic circuit models considered in this
work, then introduce some basic concepts like affine projections of a polynomial, the group
of symmetries of a polynomial and its associated Lie algebra, and finally present some known
algorithmic results like computing the derivatives of a polynomial, translation equivalence
test, variable reduction etc. In Chapter 3, we analyse the structures of the Lie algebra of
IMM and determinant polynomials. The structure of the Lie algebra of the determinant is
well-know and we present a proof for completeness. Understanding the structure of the Lie
algebra of these polynomials help us in designing an equivalence testing algorithm for them.
In Chapter 4, we present our equivalence testing algorithm for the IMM polynomial and prove
Theorems 1.1a and 1.1b. The algorithm for equivalence testing of IMM also helps us to show
that IMM is characterized by its group of symmetries, which we present in Chapter 5. In
Chapter 6, we present our algorithms for average-case LMF and average-case ABP recon-
struction and prove Theorems 1.2 and 1.3. In Chapter 7, we present our size lower bounds
on multilinear depth three and $\alpha$-balanced interval set-multilinear formulas computing IMM,
and prove Theorems 1.4 and 1.7. The contents of Section 3.1 and Chapters 4 and 5 are from
[KNST19] which is a joint work with Neeraj Kayal, Chandan Saha, and Sébastien Tavenas.
The contents of Chapter 6 are from [KNS19] which is a joint work with Neeraj Kayal, and
Chandan Saha. The contents of Section 7.2 are from [KNS16] which is a joint work with
Neeraj Kayal, and Chandan Saha.
Chapter 2

Preliminaries

In this chapter we introduce a few notations and present some well-known preliminary results which would aid us in later chapters.

Natural numbers are denoted by $\mathbb{N} = \{1, 2, \ldots \}$. Similarly $\mathbb{Q}, \mathbb{R}, \mathbb{C}$ denote the rational numbers, real numbers and complex numbers respectively. The group of invertible $n \times n$ matrices over a field $\mathbb{F}$ is denoted as $\text{GL}(n, \mathbb{F})$. Similarly the group of matrices with determinant one over a field $\mathbb{F}$ is denoted as $\text{SL}(n, \mathbb{F})$. If $\mathbb{F}$ is clear from the context then we omit $\mathbb{F}$ and write $\text{GL}(n)$ or $\text{SL}(n)$. We use $\biguplus$ to denote the disjoint union of sets.

2.1 Algebraic circuit models

Arithmetic circuits and formulas: Arithmetic circuits also called straight line programs is a natural model to compute polynomials and is defined as follows.

Definition 2.1 (Arithmetic circuit) An arithmetic circuit $\varphi$ over $x$ variables and field $\mathbb{F}$ is a directed acyclic graph, where the nodes having in-degree 0 are called leaf nodes, the nodes having out-degree 0 are called output nodes, and the remaining nodes are called internal nodes. Further, suppose $u, v$ are nodes in $\varphi$ such that there is an edge from $u$ to $v$. Then $u$ is called the child of $v$, alternatively $v$ is called the parent of $u$. The leaf nodes of $\varphi$ are labelled by an $x$ variable or by an element in $\mathbb{F}$, and every other node is either labelled by a ‘+’ or ‘×’. The edges of $\varphi$ are labelled by elements in $\mathbb{F}$. A node $v$ in $\varphi$ computes a polynomial $\varphi_v$ as follows: if $v$ is a leaf node labelled by $x \in x$ (respectively by $\alpha \in \mathbb{F}$) then $\varphi_v = x$ (respectively $\varphi_v = \alpha$), if $v$ is labelled by ‘×’ with children $v_1, v_2, \ldots, v_k$ and the edge from $v_i$ to $v$ is labelled by $\alpha_i$ for all $i \in [k]$, then $\varphi_v = \prod_{i=1}^{k} \alpha_i f_{v_i}$ and similarly if $v$ is labelled by ‘+’ with children $v_1, v_2, \ldots, v_k$ and the edge from $v_i$ to $v$ is labelled by $\alpha_i$ for all $i \in [k]$, then $\varphi_v = \sum_{i=1}^{k} \alpha_i f_{v_i}$. The polynomials computed by $\varphi$ are the polynomials computed by the output nodes of $\varphi$. 
In Definition 2.1, we have defined an arithmetic circuit that computes multiple polynomials, but usually in literature an arithmetic circuit has only a single output node. The size of an arithmetic circuit is equal to the number of nodes in it and the depth of an arithmetic circuit is equal to the length of the longest path from a leaf node to an output node. At times the size of an arithmetic circuit is taken as the number of nodes and edges in the corresponding graph. As the number of nodes and edges in a graph are polynomially related, so are these two different notions of sizes. Next we define arithmetic formulas.

**Definition 2.2 (Arithmetic formula)** An arithmetic formula is an arithmetic circuit with the out-degree of every node being one.

Alternatively an arithmetic formula is an arithmetic circuit whose underlying directed graph is a tree.

**Multilinear formulas:** A polynomial $f$ is a multilinear if the degree of every variable in it is at most one. Important polynomials like the determinant, permanent and the iterated matrix multiplication (IMM) polynomials are multilinear. Further, a degree $d$ polynomial $f$ is a set-multilinear polynomial in variable sets $x_1, \ldots, x_d$ if every monomial in it has one variable from each $x_i, i \in [d]$. Polynomials like determinant, permanent and IMM are set-multilinear. In case of determinant and permanent the variable sets correspond to variables in a single column/row. For the IMM the variable sets correspond to variables in each matrix (see Section 2.3). An arithmetic circuit/formula is multilinear if every node in it computes a multilinear polynomial. In Chapter 7 we prove an exponential separation between multilinear depth four and depth three circuits which are defined as follows.

**Definition 2.3 (Multilinear depth four and depth three circuits)** A circuit $C = \sum_{i=1}^{s} \prod_{j=1}^{d_i} Q_{ij}(x_{ij})$ is a multilinear depth four ($\Sigma \Pi \Sigma \Pi$) circuit in $x$ variables over a field $F$, if $x = \oplus_{j=1}^{d_i} x_{ij}$ for all $i \in [s]$, and $Q_{ij} \in \mathbb{F}[x_{ij}]$ is a multilinear polynomial for every $i \in [s]$ and $j \in [d_i]$. If $Q_{ij}$’s are linear polynomials then $C$ is a multilinear depth three ($\Sigma \Pi \Sigma$) circuit. The parameter $s$ is the top fan-in of $C$.

**Algebraic branching program:** In Chapter 6 we study average-case reconstruction of ABP which is a well-studied model to compute polynomials in arithmetic circuit complexity and is at least as powerful as arithmetic formulas. In Definition 2.4 and from thereon, a matrix is called a linear matrix, if all its entries are affine forms in $x$ variables.

**Definition 2.4 (Algebraic branching program)** An algebraic branching program (ABP) of width $w$, and length $d$ over a field $F$ is a product of $d$ linear matrices $X_1 \cdot X_2 \ldots X_d$, where
X₁, Xₙ are row and column vectors of length w respectively, and for k ∈ [2, d − 1], Xₖ is a w × w matrix. The entries in X₁ to Xₙ are affine forms in the variables x = {x₁, x₂, . . . , xₘ}. The polynomial computed by the ABP is the entry of the 1 × 1 matrix obtained from the product \( \prod_{k=1}^{d} X_k \). An ABP of width w, length d computing an n variate polynomial will be denoted as a (w, d, n)-ABP.

**Remarks:**

1. **Alternate definition:** A more general way to define an ABP is to consider matrices of varying dimension, that is the i-th matrix has dimension \( w_i \times w_{i+1} \), and \( w_1 = w_{d+1} = 1 \). Size of such an ABP is equal to \( \sum_{i=1}^{d+1} w_i \) and the width of the ABP is equal to \( w = (w_2, . . . , w_d) \). Alternatively, an ABP is also defined as a layered directed acyclic graph with a source s and a sink t. A length d ABP has \( d + 1 \) layers with \( w_i \) nodes in the i-th layer for \( i \in [d + 1] \). The first and the last layers contain one vertex each that is \( w_1 = w_{d+1} = 1 \) and these vertices are labelled s and t respectively. There is an edge from every vertex in layer i to every vertex in layer \( i + 1 \), for all \( i \in [d] \), and these edges between adjacent layers are labelled by affine forms in x variables. The weight of a path from s to t is the product of the edge labels in the path, and the polynomial computed by the ABP is the sum of the weights of all paths from s to t. In this case the size of the ABP is the number of nodes in the graph. It is easy to verify that the two definitions of ABP are equivalent. We use either of these definitions in our arguments later based on suitability.

2. **Linear matrix product:** The product of d, w × w linear matrices \( X_1 \cdot X_2 \cdots X_d \) with affine form entries in n variables is called a \((w, d, n)\)-matrix product. We note that in the matrix product formulation \( X_1, X_d \) are w × w linear matrices, while in the ABP formulation \( X_1, X_d \) are row and column linear matrices of length w respectively.

The concept of a full-rank ABP and the layer spaces of an ABP would be useful to us in Chapter 4.

**Definition 2.5 (Full-rank ABP)** A \((w, d, n)\)-ABP over \( \mathbb{F} \) is a full rank ABP if the \( w^2(d - 2) + 2w \) affine forms labelling the edges of the ABP are \( \mathbb{F} \)-linearly independent.

By identifying a linear form \( \sum_{i=1}^{n} a_i x_i \) with the vector \((a_1, . . . , a_n) \in \mathbb{R}^n \)(and vice versa), we can associate the following vector spaces with an ABP.
Definition 2.6 (Layer spaces of an ABP) Let \( X_1 \cdot X_2 \ldots X_d \) be a \((w,d,n)\)-ABP over \( \mathbb{F} \). Let \( X_i \) be the vector space in \( \mathbb{F}^n \) spanned by the homogeneous degree 1 parts of the affine forms\(^1\) in \( X_i \) for \( i \in [d] \); the spaces \( X_1, X_2, \ldots, X_d \) are called the layer spaces of the \((w,d,n)\)-ABP \( X_1 \cdot X_2 \ldots X_d \).

### 2.2 Linear algebra

We define some basic concepts from linear algebra with the objective of introducing the corresponding notations. In the definitions below, the vector spaces are over the field \( \mathbb{F} \).

**Definition 2.7 (Direct sum)** Let \( U, W \) be subspaces of a vector space \( V \). Then \( V \) is said to be the direct sum of \( U \) and \( W \) denoted \( V = U \oplus W \), if \( V = U + W \) and \( U \cap W = \{0\} \).

For \( U, W \) subspaces of a vector space \( V \), \( V = U \oplus W \) if and only if for every \( v \in V \) there exist unique \( u \in U \) and \( w \in W \) such that \( v = u + w \). Hence, \( \dim(V) = \dim(U) + \dim(W) \).

**Definition 2.8 (Null space)** Null space \( \mathcal{N} \) of a matrix \( M \in \mathbb{F}^{n \times n} \) is the space of all vectors \( v \in \mathbb{F}^n \), such that \( Mv = 0 \).

**Definition 2.9 (Coordinate subspace)** Let \( e_i = (0, \ldots, 1, \ldots, 0) \) be the unit vector in \( \mathbb{F}^n \) with 1 at the \( i \)-th position and all other coordinates zero. A coordinate subspace of \( \mathbb{F}^n \) is a space spanned by a subset of the \( n \) unit vectors \( \{e_1, e_2, \ldots, e_n\} \).

**Definition 2.10 (Invariant subspace)** Let \( M_1, M_2, \ldots, M_k \in \mathbb{F}^{n \times n} \). A subspace \( U \subseteq \mathbb{F}^n \) is called an invariant subspace of \( \{M_1, M_2, \ldots, M_k\} \) if \( M_i U \subseteq U \) for every \( i \in [k] \). A non-zero invariant subspace \( U \) is irreducible if there are no invariant subspaces \( U_1 \) and \( U_2 \) such that \( U = U_1 \oplus U_2 \), where \( U_1 \) and \( U_2 \) are properly contained in \( U \).

The following observation is immediate.

**Observation 2.1** If \( U \) is an invariant subspace of \( \{M_1, M_2, \ldots, M_k\} \) then for every \( M \in \mathcal{L} \) defined as \( \text{span}_\mathbb{F}\{M_1, M_2, \ldots, M_k\} \), \( M U \subseteq U \). Hence we say \( U \) is an invariant subspace of \( \mathcal{L} \), a space generated by matrices.

**Definition 2.11 (Closure of a vector)** The closure of a vector \( v \in \mathbb{F}^n \) under the action of a space \( \mathcal{L} \) spanned by a set of \( n \times n \) matrices is the smallest invariant subspace of \( \mathcal{L} \) containing \( v \).

Here, ‘smallest’ is with regard to dimension of invariant subspaces. Since intersection of two invariant subspaces is also an invariant subspace of \( \mathcal{L} \), the smallest invariant subspace of \( \mathcal{L} \) containing \( v \) is unique and is contained in every invariant subspace of \( \mathcal{L} \) containing \( v \). Algorithm 6 in Section 4.3.2 computes the closure of a given vector \( v \) under the action of \( \mathcal{L} \) whose basis is given.

\(^1\)Identify linear forms with vectors in \( \mathbb{F}^n \) as mentioned above.
2.3 Iterated matrix multiplication polynomial

Let $w = (w_1, w_2, \ldots, w_{d-1}) \subseteq \mathbb{N}^{d-1}$. Suppose $Q_1 = (x_1^{(1)}, x_2^{(1)}, \ldots, x_{w_1}^{(1)})$, $Q_d^T = (x_1^{(d)}, x_2^{(d)}, \ldots, x_{w_{d-1}}^{(d)})$ be row vectors, and for $k \in [2, d-1]$, $Q_k = (x_{ij}^{(k)})_{i \in [w_{k-1}], j \in [w_k]}$ be a $w_{k-1} \times w_k$ matrix, where for $i \in [w_1]$ $x_i^{(1)}$, for $i \in [w_{d-1}]$ $x_i^{(d)}$ and for $i \in [w_{k-1}], j \in [w_k]$ $x_{ij}^{(k)}$ are distinct variables. The iterated matrix multiplication polynomial $\text{IMM}_{w,d}$ is the entry of the $1 \times 1$ matrix obtained from the product $\prod_{i=1}^d Q_i$. When $d$ and $w$ are clear from the context, we drop the subscripts and simply represent it by $\text{IMM}$. For all $k \in [d]$, we denote the set of variables in $Q_k$ as $x_k$; Figure 2.1 depicts an ABP computing $\text{IMM}_{w,d}$ when the width is uniform, that is $w_1 = w_2 = \cdots = w_{d-1}$. If the width of the matrices are uniform and are equal to $w$ then we denote the polynomial as $\text{IMM}_{w,d}$.

Ordering of variables in $\text{IMM}_{w,d}$: We will assume that the variables $x_1 \oplus x_2 \oplus \cdots \oplus x_d$ are ordered as follows: For $i < j$, the $x_i$ variables have precedence over the $x_j$ variables. Among the $x_i$ variables, we follow column-major ordering, i.e. $x_{11}^{(l)} \succ \cdots \succ x_{w_{d-1}w_1}^{(l)} \succ \cdots \succ x_{1w_1}^{(l)} \succ x_{w_{d-1}1}^{(l)}$. We would also refer to the variables of $\text{IMM}$ as $x = \{x_1, x_2, \ldots, x_n\}$ where $x_i$ is the $i$-th variable according to this ordering, and $n = w_1 + \sum_{k=2}^{d-1} w_{k-1}w_k + w_{d-1}$ is the total number of variables in $\text{IMM}_{w,d}$.

2.4 Complexity measures

A lower bound proof on a circuit model is usually proved via a complexity measure which is a map from $\mathbb{F}[x]$ to $\mathbb{N}$. The measure is defined such that it exploits the weakness of the circuit model and it is additionally beneficial if the map is linear. We have used two complexity measures, namely evaluation dimension and a novel variant of the dimension of the space of partial derivatives we call skewed partial derivatives.
**Evaluation dimension:** This measure first defined in [FS13] is nearly equivalent variant of another measure, the *rank of the partial derivative matrix*, first defined in [Nis91] to prove lower bounds for non-commutative models. Rank of the partial derivatives matrix measure was also used in [Raz09, Raz06, RY08, RY09, DMPY12] to prove lower bounds and separations for several multilinear models. These two measures are identical over fields of characteristic zero (or sufficiently large).

**Definition 2.12 (Evaluation Dimension)** The evaluation dimension of a polynomial $g \in \mathbb{F}[x]$ with respect to a set $S \subseteq x$, denoted as $\text{Evaldim}_S(g)$, is defined as

$$\dim(\text{span}_F \{g(x) | \forall x_j \in S \ x_j = \alpha_j : \forall x_j \in S \ \alpha_j \in \mathbb{F} \}).$$

**Skewed partial derivatives:** The partial derivatives measure was introduced in [NW97]. The following is a simple variant of this measure that is also inspired by the measure used in [Nis91, Raz09].

**Definition 2.13 (Skewed partial derivatives)** Let $f \in \mathbb{F}[x, y]$, where $x$ and $y$ are disjoint sets of variables, and $Y_k$ be the set of all monomials in $y$ variables of degree $k \in \mathbb{N}$. Define the measure $\text{PD}_y(f)$ as

$$\dim \left( \text{span}_F \left\{ \left[ \frac{\partial f(x, y)}{\partial m} \right]_{\forall y \in Y, y=0} : m \in Y_k \right\} \right).$$

To prove a lower bound on the size of multilinear depth three formulas computing IMM (Theorem 1.4) we consider a polynomial $f(x, y)$ which is a projection of IMM, that is $f$ is obtained from IMM by mapping the variables in IMM to either a variable in $f$ or a field element. Since the projection preserves the multilinearity of the formula, a lower bound for $f(x, y)$ implies the same lower bound for IMM. The polynomial $f(x, y)$ is such that there is a significant difference (or skew) between the number of $x$ and $y$ variables – it is this imbalance that plays a crucial role in the proof. Both the above measures obey the property of sub-additivity.

**Lemma 2.1 (Sub-additivity)**

1. Let $g_1, g_2 \in \mathbb{F}[x]$ and $S \subseteq x$, then $\text{Evaldim}_S(g_1 + g_2) \leq \text{Evaldim}_S(g_1) + \text{Evaldim}_S(g_2)$.

2. Let $f_1, f_2 \in \mathbb{F}[x, y]$, then $\text{PD}_y(f_1 + f_2) \leq \text{PD}_y(f_1) + \text{PD}_y(f_2)$.

They attributed the notion to Ramprasad Saptharishi.
Proof: For $i \in \{1, 2\}$, let
\[
V_i = \text{span}_F \{ g_i(x) | \forall x_j \in S, x_j = \alpha_j : \forall \alpha_j \in F \} \text{ and } \\
W = \text{span}_F \{ (g_1 + g_2)(x) | \forall x_j \in S, x_j = \alpha_j : \forall \alpha_j \in F \}.
\]

Every polynomial in $W$ belongs to $V_1 + V_2$, where $V_1 + V_2 = \{ f_1 + f_2 | f_1 \in V_1, f_2 \in V_2 \}$. Hence, $\text{Evaldim}_S(g_1 + g_2) = \dim(W) \leq \dim(V_1 + V_2) \leq \dim(V_1) + \dim(V_2) = \text{Evaldim}_S(g_1) + \text{Evaldim}_S(g_2)$.

Proving part two is similar to part one. For $i \in \{1, 2\}$ let
\[
A_i = \text{span}_F \left\{ \left[ \frac{\partial f_i(x, y)}{\partial m} \right]_{\forall y \in Y, y = 0} : m \in Y_k \right\} \text{ and } \\
B = \text{span}_F \left\{ \left[ \frac{\partial (f_1 + f_2)(x, y)}{\partial m} \right]_{\forall y \in Y, y = 0} : m \in Y_k \right\}
\]

Again observing $B$ is a subspace of $A_1 + A_2$, where $A_1 + A_2 = \{ g_1 + g_2 | g_1 \in A_1, g_2 \in A_2 \}$, part two follows. ■

### 2.5 Affine and $p$-projections

**Affine projection**: Studying polynomials by applying linear transformations (from suitable matrix groups) on the variables is at the heart of invariant theory.

**Definition 2.14 (Affine projection)** An $m$-variate polynomial $f$ is an affine projection of a $n$-variate polynomial $g$, if there exists a matrix $A \in \mathbb{F}^{n \times m}$ and a $b \in \mathbb{F}^n$ such that $f(x) = g(Ax+b)$.

In [Kay12a], it was shown that given an $m$-variate polynomial $f$ and an $n$-variate polynomial $g$, checking whether $f$ is an affine projection of $g$ is NP-hard, even if $f$ and $g$ are given in the dense representation (that is as list of coefficients of the monomials). In the above definition, we say $f$ is an affine projection of $g$ via a full rank transformation, if $m \geq n$ and $A$ has rank $n$. In the affine projection via full rank transformation problem, we are given an $m$-variate polynomial $f$ and an $n$-variate polynomial $g$ in some suitable representation, and we need to determine if $f$ is an affine projection of $g$ via a full rank transformation. [Kay11, Kay12a, GGKS19] studied the affine projection via full rank transformation problem for $g$ coming from fixed families and gave polynomial time randomized algorithms to check whether a degree $d$ polynomial $f$ given as blackbox is an affine projection of $g$ via a full rank transformation, when $g$ is the elementary symmetric polynomial/permanent/determinant/power symmetric polynomial. 


polynomial or sum-of-products polynomial. In Chapter 4, we present our algorithm which checks whether a degree $d$ polynomial given as blackbox is an affine projection of IMM$_{w,d}$ via a full-rank transformation (see Theorem 1.1a).

$p$-projections: In arithmetic circuit complexity the notion of reductions is captured by the concept of $p$-projections.

**Definition 2.15 ($p$-projections)** A polynomial family $\{f_m\}_{m \in \mathbb{N}}$ is a $p$-projection of a polynomial family $\{g_n\}_{n \in \mathbb{N}}$ if there is a polynomial function $t : \mathbb{N} \to \mathbb{N}$ such that for every $m \in \mathbb{N}$ there is an $n \leq t(m)$ such that $f_m$ is an affine projection of $g_n$.

If an $m$ variate polynomial $f$ is an affine projection of an $n$ variate polynomial $g$ such that, $n = m^{O(1)}$ and $g$ is computed by a size $s$ arithmetic circuit (respectively ABP or formula), then $f$ is computed by an arithmetic circuit (respectively ABP or formula) of size $s^{O(1)}$. Thus if $\{f_m\}_{m \in \mathbb{N}}$ is a $p$-projection of $\{g_n\}_{n \in \mathbb{N}}$ and $\{g_n\}_{n \in \mathbb{N}}$ belongs to VP (respectively VBP or VF) then $\{f_m\}_{m \in \mathbb{N}}$ belongs to VP (respectively VBP or VF),

### 2.6 Group of symmetries and Lie algebra

**Definition 2.16 (Group of symmetries)** The group of symmetries of a polynomial $g \in \mathbb{F}[x]$ in $n$ variables, denoted as $G_g$, is the set of all $A \in \text{GL}(n)$ such that $g(Ax) = g(x)$.

The equivalence testing algorithm for the determinant polynomial in Theorem 6.1 uses the knowledge of the group of symmetries and its associated Lie algebra (see Definition 2.17 below). Let $\text{Det}_n$ denote the determinant of $X = (x_{ij})_{i,j \in [n]}$, where $x = \{x_{ij}\}_{i,j \in [n]}$, and let $A(X)$ denote the $n \times n$ linear matrix obtained by applying a transformation $A \in \mathbb{F}^{n^2 \times n^2}$ on $x$. Then Theorem 2.1, which states the group of symmetries of $\text{Det}_n$ is well-known (for a proof see [MM59]). Also see the references after Theorem 43 in [Kay12a] for more on $G_{\text{Det}_n}$.

**Theorem 2.1** An $A \in \text{GL}(n^2, \mathbb{F})$ is in $G_{\text{Det}_n}$ if and only if there are two matrices $S, T \in \text{SL}(n, \mathbb{F})$ such that either $A(X) = S \cdot X \cdot T$ or $A(X) = S \cdot X^T \cdot T$.

To design an equivalence test algorithm for IMM$_{w,d}$ (Theorem 1.1b) and $\text{Det}_n$, we use the structures of the Lie algebras associated with their group of symmetries. We will abuse terminology slightly and say the Lie algebra of a polynomial to mean the Lie algebra of the group of symmetries of the polynomial. We will work with the following definition of the Lie algebra of a polynomial (see [Kay12a]).

**Definition 2.17 (Lie algebra of a polynomial)** The Lie algebra of a polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ denoted as $\mathfrak{g}_f$ is the set of all $n \times n$ matrices $E = (e_{ij})_{i,j \in [n]}$ in $\mathbb{F}^{n \times n}$ such that $\sum_{i,j \in [n]} e_{ij} x_j \frac{\partial f}{\partial x_i} = 0$. 

37
Remark: Observe that $g_f$ is a subspace of $\mathbb{F}^{n \times n}$. It can also be shown that the space $g_f$ satisfies the Lie bracket property: For any $E_1, E_2 \in g_f$, $[E_1, E_2] \overset{\text{def}}{=} E_1 E_2 - E_2 E_1$ is also in $g_f$. We analyse the structure of the Lie algebra of the group of symmetries of IMM$_{w,d}$ in Section 3.1. The Lie algebra of Det$_n$ (Lemma 3.7) on the other hand is well-known but we prove it in Section 3.2.

The following fact is well-known. We provide a proof for completeness.

Claim 2.1 If $f(x) = g(Ax)$, where $f$ and $g$ are both $n$ variate polynomials and $A \in \text{GL}(n)$, then the Lie algebra of $f$ is a conjugate of the Lie algebra of $g$ via $A$, i.e. $g_f = \{ A^{-1} E A : E \in g_g \} = A^{-1} g_g A$.

Proof: Let $Q = (q_{i,j})_{i,j \in [n]} \in g_f$. Hence,

$$\sum_{i,j \in [n]} q_{i,j} x_j \cdot \frac{\partial f}{\partial x_i} = 0 \Rightarrow \sum_{i,j \in [n]} q_{i,j} x_j \cdot \frac{\partial g(Ax)}{\partial x_i} = 0 . \quad (2.1)$$

Let $A = (a_{ki})_{k,i \in [n]}$. Using chain rule of derivatives,

$$\frac{\partial g(Ax)}{\partial x_i} = \sum_{k \in [n]} \frac{\partial g}{\partial x_k}(Ax) \cdot a_{ki} .$$

Let $A^{-1} = (b_{jl})_{j,l \in [n]}$ and $(Ax)_l$ be the $l$-th entry of $Ax$. Then $x_j = \sum_{l \in [n]} b_{jl}(Ax)_l$. From Equation (2.1),

$$\sum_{i,j \in [n]} q_{i,j} \left( \sum_{l \in [n]} b_{jl}(Ax)_l \right) \cdot \left( \sum_{k \in [n]} \frac{\partial g}{\partial x_k}(Ax) \cdot a_{ki} \right) = 0 ,$$

$$\Rightarrow \sum_{k,l \in [n]} (Ax)_l \cdot \frac{\partial g}{\partial x_k}(Ax) \cdot \left( \sum_{i,j \in [n]} a_{ki} q_{i,j} b_{jl} \right) = 0 ,$$

$$\Rightarrow \sum_{k,l \in [n]} x_l \cdot \frac{\partial g}{\partial x_k} \left( \sum_{i,j \in [n]} a_{ki} q_{i,j} b_{jl} \right) = 0 \quad (\text{Substituting } x \text{ by } A^{-1} x).$$

Observe that $\sum_{i,j \in [n]} a_{ki} q_{i,j} b_{jl}$ is the $(k, l)$-th entry of $AQA^{-1}$. Hence, $AQA^{-1} \in g_g$ implying $g_f \subseteq A^{-1} g_g A$. Similarly, $g_g \subseteq Ag_f A^{-1}$ as $g = f(A^{-1} x)$, implying $g_f = A^{-1} g_g A$.

The following observation relates the invariant subspaces of the Lie algebras of two equivalent polynomials.
Observation 2.2 Suppose $f(x) = g(Ax)$, where $x = \{x_1, x_2, \ldots, x_n\}$ and $A \in \text{GL}(n)$. Then $\mathcal{U} \in \mathbb{F}^n$ is an invariant subspace of $g_y$ if and only if $A^{-1}\mathcal{U}$ is an invariant subspace of $g_f$.

Proof: $\mathcal{U}$ is an invariant subspace of $g_y$ implies, for all $E \in g_y$, $E \mathcal{U} \subseteq \mathcal{U}$. Consider $E' \in g_f$, using Claim 2.1 we know there exists $E \in g_g$ such that $AE'A^{-1} = E$. Since $\mathcal{U}$ is an invariant subspace of $AE'A^{-1}$, $A^{-1}\mathcal{U}$ is an invariant subspace of $E'$. The proof of the other direction is similar.

2.7 Algorithmic preliminaries

We record some of the basic algorithmic tasks on polynomials that can be performed efficiently and which we require at different places in our algorithms and proofs.

1. Computing homogeneous components of $f$: The $i$-th homogeneous component (or the homogeneous degree $i$ part) of a degree $d$ polynomial $f$, denoted as $f[i]$ is the sum of the degree $i$ monomials with coefficients as in $f$. Clearly, $f = f[d] + f[d-1] + \cdots + f[0]$. Given an $n$ variate, degree $d$ polynomial $f$ as a blackbox, there is an efficient algorithm to compute blackboxes for the $d$ homogeneous components of $f$. The idea is to multiply each variable by a new formal variable $t$, and then interpolate the coefficients of $t^0, t^1, \ldots, t^d$; the coefficient of $t^i$ is $f[i]$.

2. Computing derivatives of $f$: Given a polynomial $f(x_1, x_2, \ldots, x_n)$ of degree $d$ as a blackbox, we can efficiently construct blackboxes for the derivatives $\partial_{x_i} f$, for all $i \in [n]$. The following observation suggests that it is sufficient to construct blackboxes for certain homogeneous components.

Observation 2.3 If $g(x_1, x_2, \ldots, x_n)$ is a homogeneous polynomial of degree $d$ then for all $i \in [n]$

$$\partial_{x_i} g = \sum_{j=1}^{d} j \cdot x_i^{j-1} [g(x_1, x_2, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)]^{d-j}.$$  

For every $i \in [n]$, constructing a blackbox for $\partial_{x_i} f$ is immediate from the above observation as $\partial_{x_i} f = \partial_{x_i} f[d] + \partial_{x_i} f[d-1] + \cdots + \partial_{x_i} f[1]$. In fact we can use Observation 2.3 repeatedly to efficiently compute the derivative of $f$ with respect to a constant degree monomial. We record this in the following claim.

Claim 2.2 There is a deterministic algorithm that given blackbox access to an $n$ variate degree $d$ polynomial $f \in \mathbb{F}[x]$, and a monomial $\mu$ of constant degree in $x$, computes blackbox access to $\partial_{\mu} f$ in $(nd\beta)^{O(1)}$ time, where $\beta$ is the bit length of the coefficients in $f$.

3. Computing irreducible factors of a polynomial: The following result on blackbox polynomial factorization is by [KT90].
Lemma 2.2 ([KT90]) There is a randomized algorithm that takes as input blackbox access to an $n$ variate degree $d$ polynomial $f$ over $\mathbb{F}$, and constructs blackbox access to the irreducible factors of $f$ over $\mathbb{F}$ in $(nd/3)^{O(1)}$ time with success probability $1 - (nd)^{-\Omega(1)}$.

Note that the algorithm in Lemma 2.2 also works over $\mathbb{F}_q$, a finite field if the following is satisfied: If $g$ is an irreducible factor of $f$ with multiplicity $e$ then $\text{char}(\mathbb{F}) \nmid e$.

4. Space of linear dependencies of polynomials: Let $f_1, f_2, \ldots, f_m$ be $n$ variate polynomials in $\mathbb{F}[x]$ with degree bounded by $d$. The set $\mathcal{U} = \{(a_1, a_2, \ldots, a_m) \in \mathbb{F}^m \mid \sum_{j \in [m]} a_j f_j = 0\}$, called the space of $\mathbb{F}$-linear dependencies of $f_1, f_2, \ldots, f_m$ is a subspace of $\mathbb{F}^m$. We would like to find a basis of the space $\mathcal{U}$ given blackbox access to $f_1, f_2, \ldots, f_m$. Suppose the dimension of the $\mathbb{F}$-linear space spanned by the polynomials $f_1, f_2, \ldots, f_m$ is $m - r$ then $\dim(\mathcal{U}) = r$. An algorithm to find a basis of $\mathcal{U}$ can be derived from the following claim. The space $\mathcal{U}$ is equal to the null space of $M$ as defined in Claim 2.3 with high probability.

Claim 2.3 With probability at least $1 - \frac{1}{n^{\Omega(1)}}$, the rank of the matrix $M = (f_j(b_i))_{i,j \in [m]}$ is $m - r$ where $b_1, b_2, \ldots, b_m$ are chosen independently and uniformly at random from $S^m \subset \mathbb{F}^n$ with $|S| = dm \cdot n^{O(1)}$.

Proof: Recall, we assumed that the dimension of the $\mathbb{F}$-linear space spanned by the $n$ variate polynomials $f_1, f_2, \ldots, f_m$ is $m - r$. Without loss of generality assume $f_1, f_2, \ldots, f_{m-r}$ form a basis of this linear space. Clearly, the rank of $M = (f_j(b_i))_{i,j \in [m]}$ is less than or equal to $m - r$. Let $M_{m-r} = (f_j(b_i))_{i,j \in [m-r]}$. That $\det(M_{m-r}) \neq 0$ with probability at least $1 - \frac{1}{n^{\Omega(1)}}$ over the random choices of $b_1, b_2, \ldots, b_m$ can be argued as follows: Let $y_i = \{y_1^{(i)}, y_2^{(i)}, \ldots, y_n^{(i)}\}$ for $i \in [m-r]$ be disjoint sets of variables. Rename the $x = \{x_1, x_2, \ldots, x_n\}$ variables in $f_j(x)$ to $y_i$ and call these new polynomials $f_j(y_i)$ for $i, j \in [m-r]$. Let $Y$ be an $(m-r) \times (m-r)$ matrix whose $(i,j)$-th entry is $(f_j(y_i))_{i \in [m-r]}$. Since $f_1, f_2, \ldots, f_{m-r}$ are $\mathbb{F}$-linearly independent, $\det(Y) \neq 0$ – this can be argued easily using induction. As $\deg(\det(Y)) = d(m-r) \leq dm$, by Schwartz-Zippel lemma, $\det(M_{m-r}) \neq 0$ with probability at least $1 - \frac{1}{n^{\Omega(1)}}$.

5. Eliminating redundant variables:

Definition 2.18 (Essential and redundant variables) We say an $n$ variate polynomial $f$ has $s$ essential variables if there exists an $A \in \text{GL}(n)$ such that $f(Ax)$ is an $s$ variate polynomial and there exists no $A' \in \text{GL}(n)$ such that $f(A'x)$ is a $t$ variate polynomial where $t < s$. An $n$ variate polynomial has $r$ redundant variables if it has $s = n - r$ essential variables.
If the number of essential variables in a polynomial \( f(x_1, x_2, \ldots, x_n) \) is \( s \) then without loss of generality we can assume that the first \( s \) variables \( x_1, x_2, \ldots, x_s \) are essential variables and the remaining variables are redundant. An algorithm to eliminate the redundant variables of a polynomial was considered in [Car06], and it was shown that if the coefficients of a polynomial are given as input then we can eliminate the redundant variables in polynomial time. Further, [Kay11] gave an efficient randomized algorithm to eliminate the redundant variables in a polynomial given as blackbox. For completeness, we give the algorithm in [Kay11] as part of the following claim.

**Claim 2.4** Let \( r \) be the number of redundant variables in an \( n \) variate polynomial \( f \) of degree \( d \). Then the dimension of the space \( \mathcal{U} \) of \( \mathbb{F} \)-linear dependencies of \( \{ \partial_i f \mid i \in [n] \} \) is \( r \). Moreover, we can construct an \( A \in \text{GL}(n) \) in randomized \( (nd\beta)^{O(1)} \) time such that \( f(Ax) \) is free of the set of variables \( \{ x_{n-r+1}, x_{n-r+2}, \ldots, x_n \} \), where \( \beta \) is the bit length of the coefficients of \( f \).

**Proof:** Let \( B = (b_{ij})_{i,j \in [n]} \in \text{GL}(n) \) such that \( f(Bx) \) is a polynomial in \( x_1, x_2, \ldots, x_s \), where \( s = n - r \). For \( n - r + 1 \leq j \leq n \)

\[
\frac{\partial f(Bx)}{\partial x_j} = 0
\]

\[
\Rightarrow \sum_{i=1}^{n} b_{ij} \cdot \frac{\partial f}{\partial x_i}(Bx) = 0 \quad \text{(by chain rule)}
\]

\[
\Rightarrow \sum_{i=1}^{n} b_{ij} \cdot \frac{\partial f}{\partial x_i} = 0 \quad \text{(substituting } x \text{ by } B^{-1}x).}
\]

Since \( B \in \text{GL}(n) \), we conclude \( \dim(\mathcal{U}) \geq r \). Let \( \{(a_{1j} a_{2j} \ldots a_{nj})^T : (n - \dim(\mathcal{U}) + 1) \leq j \leq n\} \) be a basis of \( \mathcal{U} \). Then,

\[
\sum_{i=1}^{n} a_{ij} \cdot \frac{\partial f}{\partial x_i} = 0.
\]

Let \( A \in \text{GL}(n) \) such that for \( (n - \dim(\mathcal{U}) + 1) \leq j \leq n \), the \( j \)-th column of \( A \) is \( (a_{1j} a_{2j} \ldots a_{nj})^T \) and the remaining columns of \( A \) are arbitrary vectors that make \( A \) a full rank matrix. Then,

\[
\sum_{i=1}^{n} a_{ij} \cdot \frac{\partial f}{\partial x_i} = 0 \quad \Rightarrow \quad \sum_{i=1}^{n} a_{ij} \cdot \frac{\partial f}{\partial x_i}(Ax) = 0 \quad \Rightarrow \quad \frac{\partial f(Ax)}{\partial x_j} = 0.
\]

This implies \( f(Ax) \) is a polynomial free of \( x_j \) variable for \( (n - \dim(\mathcal{U}) + 1) \leq j \leq n \). Hence, \( \dim(\mathcal{U}) \leq r \).
Blackbox for polynomials $\partial_x f_1, \partial_x f_2, \ldots, \partial_x f_n$ can be constructed in $(nd\beta)^{O(1)}$ time from blackbox access to $f$ and a basis for the space $U$ of $\mathbb{F}$-linear dependencies of polynomials $\partial_x f_1, \partial_x f_2, \ldots, \partial_x f_n$ can also be constructed in randomized $(nd\beta)^{O(1)}$ time (see Section 2.7). Thus, we can construct an $A \in \text{GL}(n)$ (similar to the construction shown above) from a blackbox access to $f$ in randomized $(nd\beta)^{O(1)}$ time such that $f(Ax)$ is free of the set of variables $\{x_{n-r+1}, x_{n-r+2}, \ldots, x_n\}$. We summarize this in Algorithm 1.

Algorithm 1 Eliminating redundant variables

INPUT: Blackbox access to an $n$ variate polynomial $f(x)$.
OUTPUT: An $r$ and an $A \in \text{GL}(n)$ such that $r$ is the number of redundant variables in $f$ and $f(Ax)$ is free of the variables $x_{n-r+1}, x_{n-r+2}, \ldots, x_n$.

1: Compute blackbox access to $\partial_x f_1, \partial_x f_2, \ldots, \partial_x f_n$ (see Section 2.7).
2: Compute a basis $\{v_1, v_2, \ldots, v_r\}$ of the space of $\mathbb{F}$-linear dependencies of $\partial_x f_1, \partial_x f_2, \ldots, \partial_x f_n$ (using the random substitution idea in Claim 2.3). /* This step succeeds in computing the required basis with high probability. */
3: Construct an $A \in \text{GL}(n)$ such that the last $r$ columns of $A$ are $v_1, v_2, \ldots, v_r$ and the remaining columns of $A$ are chosen arbitrarily to make $A$ a full rank matrix.
4: Return $r$ and $A$.

6. Efficient translation equivalence test: Two $n$ variate degree $d$ polynomials $f, g \in \mathbb{F}[x]$ are translation equivalent (also called shift equivalent in [DdOS14]) if there exists a point $a \in \mathbb{F}^n$ such that $f(x + a) = g(x)$. Translation equivalence test takes input blackbox access to two $n$ variate polynomials $f$ and $g$, and outputs an $a \in \mathbb{F}^n$ such that $f(x + a) = g(x)$ if $f$ and $g$ are translation equivalent else outputs ‘$f$ and $g$ are not translation equivalent’. As before, let $\beta$ be the bit lengths of the coefficients of $f$ and $g$. A randomized $(nd\beta)^{O(1)}$ time algorithm is presented in [DdOS14] to test translation equivalence and find an $a \in \mathbb{F}^n$ such that $f(x + a) = g(x)$, if such an $a$ exists. Another randomized test was mentioned in [Kay12a], which we present as proof of the following lemma.

Lemma 2.3 There is a randomized algorithm that takes input blackbox access to two $n$ variate, degree $d$ polynomials $f$ and $g$, and with probability at least $1 - \frac{1}{n^{O(1)}}$ does the following: if $f$ is translation equivalent to $g$, outputs an $a \in \mathbb{F}^n$ such that $f(x + a) = g(x)$, else outputs ‘$f$ and $g$ are not translation equivalent’. The running time of the algorithm is $(nd\beta)^{O(1)}$, where $\beta$ is the bit length of the coefficients of $f$ and $g$.

Proof: We present the algorithm formally in Algorithm 2. If it succeeds in computing a point $a \in \mathbb{F}^n$ in the end (in step 20), it performs a randomized blackbox polynomial identity test
(PIT) to check whether \( f(x + a) = g(x) \) (in step 22). If \( f \) and \( g \) are not translation equivalent, this final PIT finds it with probability at least \( 1 - \frac{1}{nO(1)} \). So, for the analysis of the algorithm we can assume there is an \( a = (a_1 \ a_2 \ \ldots \ a_n)^T \in \mathbb{F}^n \) such that \( f(x + a) = g(x) \). The strategy outlined below helps to argue the correctness of Algorithm 2.

**Strategy:** Suppose \( f(x + a) = g(x) \). By equating the degree \( d \) and degree \( d - 1 \) homogeneous components of \( f \) and \( g \) we get the following equations,

\[
f^{[d]} = g^{[d]} \quad \text{and} \quad f^{[d-1]} + \sum_{i=1}^n a_i \cdot \frac{\partial f^{[d]}}{\partial x_i} = g^{[d-1]} \Rightarrow \sum_{i=1}^n a_i \cdot \frac{\partial f^{[d]}}{\partial x_i} = g^{[d-1]} - f^{[d-1]}.
\] (2.2)

Let \( f_i = \frac{\partial f^{[d]}}{\partial x_i} \) for \( i \in [n] \). Blackbox access to the homogeneous components of \( f \): \( f^{[0]}, f^{[1]}, \ldots, f^{[d]} \), the homogeneous components of \( g \): \( g^{[0]}, g^{[1]}, \ldots, g^{[d]} \) and \( f_1, f_2, \ldots, f_n \) can be constructed from blackbox access to \( f \) and \( g \) in \( (nd\beta)^{O(1)} \) time (see points 1 and 2 in Section 2.7). If \( f_1, f_2, \ldots, f_n \) are \( \mathbb{F} \)-linearly independent then with high probability over the random choices of \( b_1, b_2, \ldots, b_n \in \mathbb{F}^n \) the matrix \( (f_j(b_i))_{i,j \in [n]} \) has full rank (from Claim 2.3). Hence, we can solve for \( a_1, a_2, \ldots, a_n \) uniquely from Equation (2.2). In the general case (when \( f_1, f_2, \ldots, f_n \) may be \( \mathbb{F} \)-linearly dependent), the algorithm repeatedly applies variable reduction and degree reduction (as described below) to compute \( a \).

**Variable reduction** - We construct a transformation \( A \in \text{GL}(n) \) such that \( f^{[d]}(Ax) \) has only the essential variables \( x_1, \ldots, x_m \) (see Claim 2.4). Let \( \tilde{f} = f(Ax), \tilde{g} = g(Ax) \). It is sufficient to compute a point \( b = (b_1 \ b_2 \ \ldots \ b_n)^T \in \mathbb{F}^n \) such that \( \tilde{f}(x + b) = \tilde{g}(x) \) as

\[
\tilde{f}(x + b) = \tilde{g}(x) \Rightarrow f(Ax + Ab) = g(Ax) \Rightarrow f(x + Ab) = g(x).
\]

So we can choose \( a = Ab \). As in Equation (2.2),

\[
\tilde{f}^{[d]} = \tilde{g}^{[d]} \quad \text{and} \quad \sum_{i=1}^m b_i \cdot \frac{\partial \tilde{f}^{[d]}}{\partial x_i} = \tilde{g}^{[d-1]} - \tilde{f}^{[d-1]}.
\] (2.3)

The derivatives \( \frac{\partial x_i \tilde{f}^{[d]}}{i > m} \) are zero as \( \tilde{f}^{[d]} = f^{[d]}(Ax) \) has only the essential variables \( x_1, x_2, \ldots, x_m \). Also the polynomials \( \{ \frac{\partial x_i \tilde{f}^{[d]}}{i \in [m]} \} \) are \( \mathbb{F} \)-linearly independent (by Claim 2.4). Hence, we can solve for unique \( b_1, b_2, \ldots, b_m \) satisfying Equation (2.3) as before.
Degree reduction - To compute $b_{m+1}, b_{m+2}, \ldots, b_n$ we reduce the problem to finding a point that asserts translation equivalence of two degree $d-1$ polynomials. Let $b' = (b_1 \ b_2 \ \ldots \ b_m \ 0 \ \ldots \ 0)^T$, $\hat{f} = \hat{f}(x + b')$. Further, let $e \in \mathbb{F}^m$ such that $\hat{f}(x + e) = \tilde{g}(x)$. Then the first $m$ coordinates of $e$ must be zero\(^1\) and we can choose $b = b' + e$. We have the following equations,

$$\hat{f}^{[d]}(x + e) + (\hat{f} - \hat{f}^{[d]})(x + e) = \tilde{g}^{[d]}(x) + (\tilde{g} - \tilde{g}^{[d]})(x)$$

$$\Leftrightarrow \hat{f}^{[d]}(x + e) + (\hat{f} - \hat{f}^{[d]})(x + e) = \tilde{g}^{[d]}(x) + (\tilde{g} - \tilde{g}^{[d]})(x) \quad \text{(as } \hat{f}^{[d]} = \tilde{f}^{[d]}\text{)}.$$ 

Since $\hat{f}^{[d]}$ has only $x_1, x_2, \ldots, x_m$ variables and the first $m$ coordinates of $e$ are zero, the above statement is equivalent to

$$\hat{f}^{[d]}(x) + (\hat{f} - \hat{f}^{[d]})(x + e) = \tilde{g}^{[d]}(x) + (\tilde{g} - \tilde{g}^{[d]})(x)$$

$$\Leftrightarrow (\hat{f} - \hat{f}^{[d]})(x + e) = (\tilde{g} - \tilde{g}^{[d]})(x) \quad \text{(from Equation (2.3))}.$$

The polynomials $\hat{f} - \hat{f}^{[d]}$ and $\tilde{g} - \tilde{g}^{[d]}$ have degree at most $d - 1$ and blackboxes for these polynomials can be constructed in $(nd\beta)^{O(1)}$ time. Therefore the problem reduces to computing a point $e \in \mathbb{F}^m$ that asserts translation equivalence of two degree $(d - 1)$ polynomials.

Correctness of Algorithm 2: In steps 4-11, the algorithm carries out variable reduction and computes a part of the translation $b$ that we call $b'$ in the above argument. The remaining part of $b$ (which is the vector $e$ above) is computed by carrying out degree reduction in step 12 and then inducting on lower degree polynomials. These parts are then added appropriately in step 17, and finally an $a$ is recovered in step 20.

7. Computing basis of Lie algebra: The proof of the following lemma is given in [Kay12a], for completeness we present the proof here.

Lemma 2.4 There is a randomized algorithm which when given blackbox access to an $n$ variate degree $d$ polynomial $f$, computes a basis of $\mathfrak{g}_f$ with probability at least $1 - \frac{1}{n^{O(1)}}$ in time $(nd\beta)^{O(1)}$ where $\beta$ is the bit length of the coefficients in $f$.

Proof: Recall, the Lie algebra of $f$ is the set of all matrices $E = (e_{i,j})_{i,j \in [n]}$ such that $\sum_{i,j \in [n]} e_{ij} x_j \cdot \frac{\partial f}{\partial x_i} = 0$. Hence, $\mathfrak{g}_f$ is the space of linear dependencies of the polynomials $x_j \cdot \frac{\partial f}{\partial x_i}$ for $i, j \in [n]$. Using Observation 2.3, we can derive blackboxes for these $n^2$ polynomials and then compute a basis of the space of linear dependencies with high probability using Claim 2.3. \(\blacksquare\)

\(^1\)as $b_1, b_2, \ldots, b_m$ can be solved uniquely
7. Solutions of polynomial equations: Let $g_1, \ldots, g_m \in \mathbb{F}[x]$ be $n$ variate, degree $d$ polynomials. Then the set $I \overset{\text{def}}{=} \{ \sum_{i=1}^{m} h_i g_i \mid h_1, \ldots, h_m \in \mathbb{F}[x] \}$ is called the ideal generated by $g_1, g_2, \ldots, g_m$ in $\mathbb{F}[x]$. Further, let $\overline{\mathbb{F}}$ denote the algebraic closure of $\mathbb{F}$. Then the set $V_{\overline{\mathbb{F}}}(I) \overset{\text{def}}{=} \{ a \in \overline{\mathbb{F}}^n \mid g_i(a) = 0 \text{ for all } i \in [m] \}$ is called the variety or the algebraic set defined by $I$ over $\overline{\mathbb{F}}$. $V_{\overline{\mathbb{F}}}(I)$ is zero-dimensional if it has finitely many points. We say a point $a \in V_{\overline{\mathbb{F}}}(I)$ is $\mathbb{F}$-rational if $a \in \mathbb{F}^n$. The proof of the next result follows from [Ier89] (see also [HW99]).

**Lemma 2.5 ([Ier89])** There is a randomized algorithm that takes input $m, n$ variate, degree $d$ polynomials $g_1, g_2, \ldots, g_m$ generating an ideal $I$ of $\mathbb{F}[x]$. If $V_{\overline{\mathbb{F}}}(I)$ is zero-dimensional and all points in it are $\mathbb{F}$-rational then the algorithm computes all the points in $V_{\overline{\mathbb{F}}}(I)$ with probability $1 - \exp(-mdn \log q)$. The running time of the algorithm is $(mdn \log q)^{O(1)}$.

A similar result, but for homogeneous $g_1, \ldots, g_m$, follows from [Laz01].
Algorithm 2 Translation equivalence test

INPUT: Blackbox access to two $n$ variate, degree $d$ polynomials $f$ and $g$.
OUTPUT: A point $a \in \mathbb{F}^n$ such that $f(x + a) = g(x)$, if such an $a$ exists.

1: Set $\ell = d$, $p = f$ and $q = g$.
2: 
3: while $\ell > 0$ do 
4: Using Algorithm 1 find an $m$ and an $A_\ell \in \text{GL}(n)$ such that the variables $x_{m+1}, x_{m+2}, \ldots, x_n$ do not appear in $p^{[\ell]}(A_\ell x)$. /* With high probability $m$ is the number of essential variables in $p^{[\ell]}$. */
5: Let $\tilde{p} = p(A_\ell x)$ and $\tilde{q} = q(A_\ell x)$. Construct blackbox access to $\tilde{p}^{[\ell]}, \tilde{p}^{[\ell-1]}, \tilde{q}^{[\ell]}, \tilde{q}^{[\ell-1]}$ and $\partial_{x_i} \tilde{p}^{[\ell]}$ for $i \in [m]$.
6: Check if $\tilde{p}^{[\ell]} = \tilde{q}^{[\ell]}$. If not, output ‘$f$ and $g$ are not translation equivalent’ and stop. /* The check succeeds with high probability. */
7: Solve for unique $b_1, b_2, \ldots, b_m$ satisfying
\[
\sum_{i=1}^{m} b_i \cdot \frac{\partial \tilde{p}^{[\ell]}}{\partial x_i} = \tilde{q}^{[\ell-1]} - \tilde{p}^{[\ell-1]} \quad \text{(using the random substitution idea in Claim 2.3).}
\]
   If the solving fails, output ‘$f$ and $g$ are not translation equivalent’. /* This step succeeds with high probability if $m$ is the number of essential variables in $p^{[\ell]}$ in step 4. */
8: if $m = n$ then
9:   Set $b_\ell = (b_1, b_2 \ldots b_n)$ and exit while loop.
10: else
11:   Set $b_\ell = (b_1, b_2 \ldots b_m, 0 \ldots 0) \in \mathbb{F}^n$.
12:   Construct blackbox access to $(\tilde{p} - \tilde{p}^{[\ell]})(x + b_\ell)$ and $(\tilde{q} - \tilde{q}^{[\ell]})(x)$. Set $p = (\tilde{p} - \tilde{p}^{[\ell]})(x + b_\ell)$, $q = (\tilde{q} - \tilde{q}^{[\ell]})(x)$ and $\ell = \ell - 1$.
13: end if
14: end while
15: 
16: while $\ell < d$ do
17:   Set $b_{\ell+1} = b_{\ell+1} + A_\ell b_\ell$.
18:   Set $\ell = \ell + 1$.
19: end while
20: Set $a = A_d b_d$.
21: 
22: Pick a point $c$ uniformly at random from $S^n \subset \mathbb{F}^n$ with $|S| = d.n^{O(1)}$ and check whether $f(c + a) = g(c)$. /* With high probability $f(c + a) \neq g(c)$ if $f$ and $g$ are not translation equivalent. */
23: if $f(c + a) = g(c)$ then
24:   Output the point $a$.
25: else
26:   Output ‘$f$ and $g$ are not translation equivalent’.
27: end if
Chapter 3

Lie algebra of the IMM and the determinant polynomials

In this chapter we describe the structure of the Lie algebras of the iterated matrix multiplication and the determinant polynomials. In the case of $\text{IMM}_{w,d}$, we also describe all the possible irreducible invariant subspaces of its Lie algebra. The understanding of the structure of the Lie algebras of these polynomials helps us in proving the results presented in the next few chapters. The contents of Section 3.1 are from our work [KNST19]. Lemma 3.7 in Section 3.2 is well-known (see [Kay12a]), we provide a proof for completeness.

3.1 Lie algebra of IMM

Dropping the subscripts $w$ and $d$, we refer to $\text{IMM}_{w,d}$ as IMM. We show that the Lie algebra, $\mathfrak{g}_{\text{IMM}}$, consists of well-structured subspaces and by analysing these subspaces we are able to identify all the irreducible invariant subspaces of $\mathfrak{g}_{\text{IMM}}$.

3.1.1 Structure of the Lie algebra $\mathfrak{g}_{\text{IMM}}$

Recall that $x = x_1 \oplus x_2 \oplus \cdots \oplus x_d$ are the variables of IMM which are also referred to as $\{x_1, x_2, \ldots, x_n\}$ for notational convenience. The $x$ variables are ordered as mentioned in Section 2.3.

Lemma 3.1 Let $W_1, W_2, W_3$ be the following sets (spaces) of matrices:

1. $W_1$ consists of all matrices $D = (d_{ij})_{i,j \in [n]}$ such that $D$ is diagonal and

$$\sum_{i=1}^{n} d_{ii}x_i \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0.$$
2. $W_2$ consists of all matrices $B = (b_{ij})_{i,j \in [n]}$ such that

$$
\sum_{i,j \in [n]} b_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0,
$$

where in every summand $b_{ij} \neq 0$ only if $x_i \neq x_j$ and $x_i, x_j \in x_l$ for some $l \in [d]$.

3. $W_3$ consists of all matrices $C = (c_{ij})_{i,j \in [n]}$ such that

$$
\sum_{i,j \in [n]} c_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0,
$$

where in every summand $c_{ij} \neq 0$ only if either $x_i \in x_2$, $x_j \in x_1$ or $x_i \in x_{d-1}$, $x_j \in x_d$.

Then $\mathfrak{g}_{\text{IMM}} = W_1 \oplus W_2 \oplus W_3$.

Before proving Lemma 3.1 we show how it helps us to almost pin down the structure of an element $E \in \mathfrak{g}_{\text{IMM}}$.

**Elaboration on Lemma 3.1:** An element $E = (e_{ij})_{i,j \in [n]}$ of $\mathfrak{g}_{\text{IMM}}$ is an $n \times n$ matrix with rows and columns indexed by variables of IMM following the ordering mentioned in Section 2.3. Since $\sum_{i,j \in [n]} e_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0$, $E$ appears as shown in Figure 3.1, where the row indices correspond to derivatives and column indices correspond to shifts\(^1\). The proof will show that $E$ is a sum of three matrices $D \in W_1$, $B \in W_2$ and $C \in W_3$ such that

1. $D$ contributes to the diagonal entries.

2. $B$ contributes to the block-diagonal entries of $E$ corresponding to the locations:
   
   • $(x_i^{(1)}, x_j^{(1)})$ where $i, j \in [w_1]$ and $i \neq j$
   • $(x_i^{(d)}, x_j^{(d)})$ where $i, j \in [w_{d-1}]$ and $i \neq j$
   • $(x_{ij}^{(l)}, x_{pq}^{(l)})$ where $i, p \in [w_{l-1}]$ and $j, q \in [w_l]$ for $l \in [2, d-1]$, and $(i, j) \neq (p, q)$.

3. $C$ contributes to the two corner rectangular blocks corresponding to:
   
   • rows labelled by $x_2$ variables and columns labelled by $x_1$ variables
   • rows labelled by $x_{d-1}$ variables and columns labelled by $x_d$ variables.

\(^1\)Borrowing terminology from the shifted partial derivatives measure [Kay12b].
Proof of Lemma 3.1: Since \( W_1 \cap W_2 = (W_1 + W_2) \cap W_3 = \{0_n\} \), where \( 0_n \) is the \( n \times n \) all zero matrix, it is sufficient to show \( g_{\text{MM}} = W_1 + W_2 + W_3 \). By definition, \( W_1 + W_2 + W_3 \subseteq g_{\text{MM}} \).

We now show that \( g_{\text{MM}} \subseteq W_1 + W_2 + W_3 \). Let \( E = (e_{ij})_{i,j \in [n]} \) be a matrix in \( g_{\text{MM}} \). Then \( \sum_{i,j \in [n]} e_{ij} x_j \cdot \frac{\partial \text{MM}}{\partial x_i} = 0 \). We focus on a term \( x_j \cdot \frac{\partial \text{MM}}{\partial x_i} \) and observe the following:

(a) If \( x_i = x_j \) then the monomials of \( x_j \cdot \frac{\partial \text{MM}}{\partial x_i} \) are also monomials of IMM. Such monomials do not appear in any term \( x_j \cdot \frac{\partial \text{MM}}{\partial x_i} \), where \( x_i \neq x_j \).

(b) If \( x_i \neq x_j \) and \( x_i, x_j \) belong to the same \( x_l \) then every monomial in \( x_j \cdot \frac{\partial \text{MM}}{\partial x_i} \) has exactly one variable from every \( x_k \) for \( k \in [d] \). Such monomials do not appear in a term \( x_j \cdot \frac{\partial \text{MM}}{\partial x_i} \), where \( x_i \in x_l \) and \( x_j \in x_k \) and \( l \neq k \).
Due to this monomial disjointness, an equation \( \sum_{i,j} e_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0 \) corresponding to \( E \) can be split into three equations:

1. \( \sum_{i=1}^n d_{ii} x_i \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0 \).
2. \( \sum_{i,j \in [n]} b_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0 \), where \( b_{ij} \neq 0 \) in a term only if \( x_i \neq x_j \) and \( x_i, x_j \in x_l \) for some \( l \in [d] \).
3. \( \sum_{i,j \in [n]} c_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0 \), where \( c_{ij} \neq 0 \) in a term only if \( x_i \in x_l \) and \( x_j \in x_k \) for \( l \neq k \).

Hence every \( E = (e_{ij})_{i,j \in [n]} \) in \( g_{\text{IMM}} \) equals \( D + B + C \) where

- \( D \in \mathcal{W}_1 \) is a diagonal matrix,
- \( B \in \mathcal{W}_2 \) is a block-diagonal matrix with diagonal entries zero,
- \( C \) is a matrix with nonzero entries appearing outside the above block-diagonal.

To complete the proof of the lemma we show the following.

**Claim 3.1** Except those entries of \( C \) whose rows and columns are indexed by \( x_2 \) and \( x_1 \) variables respectively, or \( x_{d-1} \) and \( x_d \) variables respectively, all the other entries are zero.

**Proof:** In a term \( x_{(l)}^{pq} \cdot \frac{\partial \text{IMM}}{\partial x_{ij}^{(k)}} \) where \( l \neq k \), every monomial has two variables from \( x_l \) and no variable from \( x_k \). Hence from the equation corresponding to \( C \) we get separate equations for every pair \( (l, k) \) due to monomial disjointness:

\[
\sum_{p \in [w_{l-1}], q \in [w_l]} \sum_{i \in [w_{k-1}], j \in [w_k]} c_{pq,ij} x_{(l)}^{pq} \cdot \frac{\partial \text{IMM}}{\partial x_{ij}^{(k)}} = 0, \quad \text{where } l \neq k.
\]

Collecting coefficients corresponding to \( \frac{\partial \text{IMM}}{\partial x_{ij}^{(k)}} \) in the above equation we get

\[
\sum_{i \in [w_{k-1}], j \in [w_k]} \ell_{ij}^{(k)} \cdot \frac{\partial \text{IMM}}{\partial x_{ij}^{(k)}} = 0, \quad \text{where } \ell_{ij}^{(k)} \text{ is a linear form in the variables from } x_l. \quad (3.1)
\]

Figure 3.2 depicts a term \( \ell_{ij}^{(k)} \cdot \frac{\partial \text{IMM}}{\partial x_{ij}^{(k)}} \) using an ABP that computes it. So the LHS of the above equation can be computed by an ABP \( B \) that has edge labels identical to that of the ABP for \( \text{IMM} \), except for the edges in layer \( k \). The \((i,j)\)-th edge of layer \( k \) in \( B \) is labelled by \( \ell_{ij}^{(k)} \).

\(^1\)An entry is in the block-diagonal if and only if the variables labelling the row and column of the entry are in the same \( x_l \) for some \( l \in [d] \).
Figure 3.2: An ABP computing the term $\ell^{(k)}_{ij} \cdot \partial \text{IMM} / \partial x^{(k)}_{ij}$

pose $\ell^{(k)}_{ij} \neq 0$ and the coefficient of the variable $x^{(l)}_{pq}$ in $\ell^{(k)}_{ij}$ is nonzero, i.e. $c_{pq,ij} \neq 0$. If $(l,k)$ is neither $(1,2)$ nor $(d,d-1)$ then the assumption $c_{pq,ij} \neq 0$ leads to a contradiction as follows.

Consider an $s$ to $t$ path $P$ in $B$ that goes through the $(i,j)$-th edge of layer $k$ (which is labelled by $\ell^{(k)}_{ij}$) but excludes the $(p,q)$-th edge of layer $l$ (which is labelled by $x^{(l)}_{pq}$), the $(p,i)$-th edge of layer $k-1$ if $l = k-1$ and the $(j,q)$-th edge of layer $k+1$ if $l = k+1$ (we can notice this is always possible since $(l,k)$ is neither $(1,2)$ nor $(d,d-1)$). Then, if we retain the variables labelling the edges of $P$ outside the layer $k$ and the variable $x^{(l)}_{pq}$, and set every other variable to zero then $P$ becomes the unique $s$ to $t$ path in $B$ with nonzero weight (since $c_{pq,ij} \neq 0$). But this contradicts the fact that ABP $B$ is computing an identically zero polynomial (by Equation (3.1)).

Therefore, $g_{\text{IMM}} \subseteq W_1 + W_2 + W_3$ implying $g_{\text{IMM}} = W_1 \oplus W_2 \oplus W_3$.

In order to get a finer understanding of $g_{\text{IMM}}$ and its dimension we look at the spaces $W_1, W_2$ and $W_3$ closely, and henceforth call them the diagonal space, the block-diagonal space and the corner space respectively.

Corner space $W_3$:

Lemma 3.2 (Corner space) The space $W_3 = W_3^{(a)} \oplus W_3^{(b)}$ where $W_3^{(a)} = A_1 \oplus A_2 \oplus \cdots \oplus A_w$ and $W_3^{(b)} = A_1' \oplus A_2' \oplus \cdots \oplus A_{w_{d-2}}'$ such that for every $i \in [w_2]$ $A_i$ is isomorphic to the space of $w_1 \times w_1$ anti-symmetric matrices over $F$, and for every $j \in [w_{d-2}]$ $A_j'$ is isomorphic to the space of $w_{d-1} \times w_{d-1}$ anti-symmetric matrices over $F$. Hence $\dim(W_3) = \frac{1}{2} w_1 w_2 (w_1 - 1) + w_{d-1} w_{d-2} (w_{d-1} - 1)$.

We briefly elaborate on the statement of the lemma before presenting the proof.

Elaboration on Lemma 3.2: Every element $C \in W_3$ can be expressed as a sum of two
$n \times n$ matrices $C^{(a)} \in \mathcal{W}^{(a)}_3$ and $C^{(b)} \in \mathcal{W}^{(b)}_3$. $C^{(a)}$ looks as shown in Figure 3.3, where for every $i \in [w_2]$ $C^{(a)}_i$ is an anti-symmetric matrix. The structure of $C^{(b)}$ is similar\(^1\) to that of $C^{(a)}$ with non zero entries restricted to the rows indexed by $x_{d-1}$ variables and columns indexed by $x_d$ variables.

![Figure 3.3: A matrix $C^{(a)}$ in $\mathcal{W}^{(a)}_3$](image)

**Proof of Lemma 3.2:** Recall, $\mathcal{W}_3$ is the space of all matrices $C = (c_{ij})_{i,j \in [n]}$ such that

$$\sum_{i,j \in [n]} c_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0, \quad (3.2)$$

where in every nonzero summand either $x_i \in x_2, x_j \in x_1$ or $x_i \in x_{d-1}, x_j \in x_d$. In Equation (3.2) every monomial in a term $x_p^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_q^{(2)}}$ has two variables from $x_1$. Similarly, every monomial in a term $x_p^{(d)} \cdot \frac{\partial \text{IMM}}{\partial x_q^{(d-1)}}$ has two variables from $x_d$ respectively. Owing to monomial disjointness, Equation (3.2) gives two equations

$$\sum_{r \in [w_2]} \sum_{p,q \in [w_1]} c^{(1)}_{pqr} x_p \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(1)}} = 0, \quad \text{and} \quad (3.3)$$

$$\sum_{q \in [w_{d-2}]} \sum_{p,r \in [w_{d-1}]} c^{(d)}_{pqr} x_p \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(d-1)}} = 0. \quad (3.4)$$

Thus $\mathcal{W}_3 = \mathcal{W}^{(a)}_3 \oplus \mathcal{W}^{(b)}_3$ where $\mathcal{W}^{(a)}_3$ consists of matrices satisfying Equation (3.3) and $\mathcal{W}^{(b)}_3$ consists of matrices satisfying Equation (3.4). We argue the following about $\mathcal{W}^{(a)}_3$.

\(^1\) once we rearrange the rows in $C^{(b)}$ indexed by variables in $x_{d-1}$ according to row major ordering (instead of column major ordering) of variables in $x_{d-1}$.
Claim 3.2 \( W^{(a)}_3 = A_1 \oplus A_2 \oplus \cdots \oplus A_{w_2} \) where every \( A_i \) is isomorphic to the space of \( w_1 \times w_1 \) anti-symmetric matrices over \( F \).

Proof: Figure 3.4 depicts an ABP computing the term \( x_p^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(2)}} \). Every monomial in \( c_{pqr}^{(1)} x_p^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(2)}} \) is divisible by \( x_p^{(1)} x_q^{(1)} \). The only other term in Equation (3.3) that contains monomials divisible by \( x_p^{(1)} x_q^{(1)} \) is \( c_{qpr}^{(1)} x_p^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(2)}} \). Figure 3.5 depicts an ABP computing \( x_q^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_{pr}^{(2)}} \). Since the terms in Figures 3.4 and 3.5 have no monomials in common with any other term in Equation (3.3) it must be that \( c_{pqr}^{(1)} = -c_{qpr}^{(1)} \). Moreover, if \( p = q \) then \( c_{pqr}^{(1)} = 0 \). Thus Equation (3.3) gives an equation for every \( r \in [w_2] \)

\[
\sum_{p,q \in [w_1], p \neq q} c_{pqr}^{(1)} x_p^{(1)} \cdot \frac{\partial \text{IMM}}{\partial x_{qr}^{(2)}} = 0,
\]

(3.5)

such that the matrix \( C_r = (c_{pqr}^{(1)})_{p,q \in [w_1]} \in F^{w_1 \times w_1} \) is anti-symmetric. Further any anti-symmetric matrix can be used to get an equation like Equation (3.5). Thus, as shown in Figure 3.6, every matrix \( C^{(a)} \in W^{(a)}_3 \) is such that for every \( r \in [w_2] \), the \( w_1 \times w_1 \) submatrix (say \( C^{(a)}_r \)) defined by the rows labelled by the \( x_{qr}^{(2)} \) variables and the columns labelled by the \( x_p^{(1)} \) variables for \( p, q \in [w_1] \) is anti-symmetric. Also, any matrix satisfying the above properties belongs to \( W^{(a)}_3 \).
Naturally, if we define $A_r$ to be the space of $n \times n$ matrices such that the $w_1 \times w_1$ submatrix defined by the rows labelled by the $x_{qr}^{(2)}$ variables and the columns labelled by the $x_p^{(1)}$ variables for $p, q \in [w_1]$ is anti-symmetric and all other entries are zero then $W_3^{(a)} = A_1 \oplus A_2 \oplus \cdots \oplus A_{w_2}$.  

Similarly, it can be shown that $W_3^{(b)} = A_1' \oplus A_2' \oplus \cdots \oplus A_{w_d-2}'$ where every $A_i'$ is isomorphic to the space of $w_d-1 \times w_d-1$ anti-symmetric matrices. This completes the proof of Lemma 3.2.  

**Block-diagonal space $W_2$:** In the following lemma, $Z_{w_k}$ denotes the space of $w_k \times w_k$ matrices with diagonal entries zero for $k \in [d-1]$. Also, for notational convenience we assume that $w_0 = w_d = 1$. We will also use the tensor product of matrices: if $A = (a_{i,j}) \in \mathbb{F}^{r \times s}$ and $B \in \mathbb{F}^{t \times u}$, then $A \otimes B$ is the $(rt) \times (su)$ matrix given by

$$A \otimes B = \begin{bmatrix} a_{1,1}B & \cdots & a_{1,s}B \\ \vdots & \ddots & \vdots \\ a_{r,1}B & \cdots & a_{r,s}B \end{bmatrix}. $$

**Lemma 3.3 (Block-diagonal space)** The space $W_2 = B_1 \oplus B_2 \oplus \cdots \oplus B_{d-1}$ such that for every $k \in [d-1]$, $B_k$ is isomorphic to the $\mathbb{F}$-linear space spanned by $t_k \times t_k$ matrices of the form

$$\begin{bmatrix} -Z^T \otimes I_{w_{k-1}} & 0 \\ 0 & I_{w_{k+1}} \otimes Z \end{bmatrix}_{t_k \times t_k},$$

where $Z \in Z_{w_k}$ and $t_k = w_k(w_{k-1} + w_{k+1})$.  

(3.6)
Hence, \( \dim(W_2) = \sum_{k=1}^{d-1}(w_k^2 - w_k) \).

The proof is presented after we briefly state the structure of the matrices in \( W_2 \).

**Elaboration on Lemma 3.3:** An element \( B \in W_2 \) is a sum of \( d - 1 \), \( n \times n \) matrices \( B_1, B_2, \ldots, B_{d-1} \) such that for every \( k \in [d - 1] \), \( B_k \in \mathcal{B}_k \) and the non zero entries of \( B_k \) are restricted to the rows and columns indexed by \( x_k \cup x_{k+1} \) variables. The submatrix in \( B_k \) corresponding to these rows and columns looks as shown in Equation (3.6).

**Proof of Lemma 3.3** Recall \( w_0 = w_d = 1 \) and \( \mathcal{Z}_{w_k} \) denotes the space of \( w_k \times w_k \) matrix with diagonal entries 0, and \( W_2 \) is the space of all matrices \( B = (b_{ij})_{i,j \in [n]} \) such that

\[
\sum_{i,j \in [n]} b_{ij} x_j \cdot \frac{\partial \text{IMM}}{\partial x_i} = 0, \tag{3.7}
\]

where in every term \( b_{ij} \neq 0 \) only if \( x_i \neq x_j \) and \( x_i, x_j \in x_l \) for some \( l \in [d] \). The following observation is easy to verify.

**Observation 3.1** Suppose \( l \in [2, d - 1] \). A term \( x_{i_1 j_1}^{(l)} \cdot \frac{\partial \text{IMM}}{\partial x_{i_2 j_2}^{(l)}} \) where \( i_1 \neq i_2 \) and \( j_1 \neq j_2 \) does not share a monomial with any other term in Equation (3.7).

Hence for \( l \in [2, d - 1] \), terms of the kind \( x_{i_1 j_1}^{(l)} \cdot \frac{\partial \text{IMM}}{\partial x_{i_2 j_2}^{(l)}} \) where \( i_1 \neq i_2 \) and \( j_1 \neq j_2 \) are absent in Equation (3.7). A monomial appearing in a non zero term of Equation (3.7) is of the form \( x_{i_1}^{(1)} \cdot x_{i_2}^{(2)} \cdots x_{k}^{(k-1)} \cdot x_{i_{k+1}}^{(k+1)} \cdots x_{i_d}^{(d-1)} \cdot x_{i_d}^{(d)} \) where \( i_k \neq i_k' \), for some \( k \in [d - 1] \). We say such a monomial is broken at the \( k \)-th interface. Observe the following.

**Observation 3.2** The terms \( x_{i}^{(k)} \cdot \frac{\partial \text{IMM}}{\partial x_{q}^{(k)}} \) where \( p \in [w_{k-1}], q, r \in [w_k], q \neq r \), and \( x_{i}^{(k+1)} \cdot \frac{\partial \text{IMM}}{\partial x_{r}^{(k+1)}} \) where \( i, m \in [w_k], j \in [w_{k+1}], i \neq m \) are the only two whose monomials are broken at the \( k \)-th interface.

Thus from Equation (3.7) we get \((d - 1)\) equations one for each interface by considering cancellations of monomials broken at that interface. For \( k \in [2, d - 2] \), let \( \mathcal{B}_k \) be the space of all \( n \times n \) matrices \( B_k \) such that

1. the entry corresponding to the row labelled by \( x_{p}^{(k)} \) and the column labelled by \( x_{p}^{(k)} \) is \( b_{pq,qr}^{(k)} \in F \) for \( p \in [w_{k-1}], q, r \in [w_k] \) and \( q \neq r \),

2. the entry corresponding to the row labelled by \( x_{i}^{(k+1)} \) and the column labelled by \( x_{i}^{(k+1)} \) is \( b_{ij,mj}^{(k+1)} \in F \) for \( i, m \in [w_k], j \in [w_{k+1}] \) and \( i \neq m \),

55
3. all other entries of $B_k$ are zero, and

4. 

$$
\sum_{p \in [w_{k-1}], q,r \in [w_k], q \neq r} b_{pq,pr}^{(k)} x_{pr}^{(k)} \cdot \frac{\partial \text{IMM}}{\partial x_{pq}^{(k)}} + \sum_{i,m \in [w_k], j \in [w_{k+1}], i \neq m} b_{ij,mj}^{(k+1)} x_{mj}^{(k+1)} \cdot \frac{\partial \text{IMM}}{\partial x_{ij}^{(k+1)}} = 0.
$$

(3.8)

We can define spaces $\mathcal{B}_1$ and $\mathcal{B}_{d-1}$ similarly considering monomials broken at the first and the last interface respectively. As Equation (3.7) can be split into $(d - 1)$ equations, one for every interface, $\mathcal{W}_2 = \mathcal{B}_1 + \mathcal{B}_2 + \cdots + \mathcal{B}_{d-1}$. Since the spaces $\mathcal{B}_1, \ldots, \mathcal{B}_{d-1}$ control different entries of $n \times n$ matrices, $\mathcal{W}_2 = \mathcal{B}_1 \oplus \mathcal{B}_2 \oplus \cdots \oplus \mathcal{B}_{d-1}$.

**Claim 3.3** For $k \in [2, d-2]$, $\mathcal{B}_k$ is isomorphic to the $\mathbb{F}$-linear space spanned by $t_k \times t_k$ matrices of the form

$$
\begin{bmatrix}
-Z \otimes I_{w_{k-1}} & 0 \\
0 & I_{w_{k+1}} \otimes Z
\end{bmatrix}_{t_k \times t_k}
$$

where $Z \in \mathbb{Z}_{w_k}$ and $t_k = w_k(w_{k-1} + w_{k+1})$.

**Proof:** Collecting same derivative terms in Equation (3.8) we get

$$
\sum_{p \in [w_{k-1}], q \in [w_k]} \ell_{pq}^{(k)} \cdot \frac{\partial \text{IMM}}{\partial x_{pq}^{(k)}} + \sum_{i \in [w_k], j \in [w_{k+1}]} \ell_{ij}^{(k+1)} \cdot \frac{\partial \text{IMM}}{\partial x_{ij}^{(k+1)}} = 0,
$$

(3.9)

where $\ell_{pq}^{(k)}$ is a linear form containing variables $x_{pr}^{(k)}$ such that $r \neq q$, and $\ell_{ij}^{(k+1)}$ is a linear form containing variables $x_{mj}^{(k+1)}$ such that $m \neq i$. Here is a succinct way to write Equation (3.9):

$$
Q_1 \cdot Q_2 \cdots Q_j' \cdot Q_{k+1} \cdot Q_{k+2} \cdots Q_{d-1} \cdot Q_d + Q_1 \cdot Q_2 \cdots Q_k \cdot Q_{k+1}' \cdot Q_{k+2} \cdots Q_{d-1} \cdot Q_d = 0,
$$

(3.10)

where $Q_1, \ldots, Q_d$ are matrices as in Section 2.3, $Q_k' = (\ell_{pq}^{(k)})_{p \in [w_{k-1}], q \in [w_k]}$ and $Q_{k+1}' = (\ell_{ij}^{(k+1)})_{i \in [w_k], j \in [w_{k+1}]}$. This implies

$$
Q_k' \cdot Q_{k+1} + Q_k \cdot Q_{k+1}' = 0,
$$

as $Q_1, \ldots, Q_d$ have distinct sets of variables, and the variables appearing in $Q_k'$ and $Q_{k+1}'$ are the same as in $Q_k$ and $Q_{k+1}$ respectively. The variable disjointness of $Q_k$ and $Q_{k+1}$ can be exploited to infer $Q_{k+1}' = Z \cdot Q_k$ and $Q_k' = -Q_k \cdot Z$ where $Z$ is in $\mathbb{F}^{w_k \times w_k}$ (even if $Q_k, Q_{k+1}$ may not be square matrices). As the linear form $\ell_{pq}^{(k)}$ is devoid of the variable $x_{pr}^{(k)}$, it must be that $Z \in \mathbb{Z}_{w_k}$. Moreover, any $Z \in \mathbb{Z}_{w_k}$ can be used along with the relations $Q_{k+1}' = Z \cdot Q_{k+1}$
and \( Q'_k = -Q_k \cdot Z \) to satisfy Equation (3.10) and hence also Equations (3.8) and (3.9).

Let \( Z = (z_{im})_{i,m \in [w_k]} \). Since \( Q'_{k+1} = Z \cdot Q_{k+1} \), the coefficient of \( x_{mj}^{(k+1)} \) in \( \ell_{ij}^{(k+1)} \) is \( z_{im} \) for every \( j \in [w_{k+1}] \). Hence in Equation (3.8), \( b^{(k+1)}_{ij,mj} = z_{im} \) for every \( j \in [w_{k+1}] \).

Similarly, since \( Q'_k = -Q_k \cdot Z \) the coefficient of \( x_{pr}^{(k)} \) in \( \ell_{pq}^{(k)} \) is \( -z_{rq} \) for every \( p \in [w_{k-1}] \). Hence in Equation (3.8) \( b^{(k)}_{pq,pr} = -z_{rq} \) for every \( p \in [w_{k-1}] \).

Thus the submatrix of \( B_k \) defined by the rows and columns labelled by the variables in \( x_k \) and \( x_{k+1} \) looks like

\[
\begin{bmatrix}
-Z^T \otimes I_{w_k-1} & 0 \\
0 & I_{w_{k+1}} \otimes Z
\end{bmatrix}
t_k \times t_k
\]

where \( t_k = w_k(w_{k-1} + w_{k+1}) \) and all other entries in \( B_k \) are zero. Hence \( B_k \) is isomorphic to the space generated by \( t_k \times t_k \) matrices of the above kind. This proves the claim.

We can similarly show that \( B_1 \) is isomorphic to the space generated by square matrices of the form

\[
\begin{bmatrix}
-Z^T & 0 \\
0 & I_{w_2} \otimes Z
\end{bmatrix}
t_1 \times t_1
\]

where \( Z \in \mathbb{Z}_{w_1} \) and \( t_1 = w_1 + w_1 w_2 \),

and \( B_{d-1} \) is isomorphic to the space generated by square matrices of the form

\[
\begin{bmatrix}
-Z^T \otimes I_{w_{d-2}} & 0 \\
0 & Z
\end{bmatrix}
t_{d-1} \times t_{d-1}
\]

where \( Z \in \mathbb{Z}_{w_{d-1}} \) and \( t_{d-1} = w_{d-1}w_{d-2} + w_{d-1} \).

This completes the proof of Lemma 3.3.

Diagonal space \( W_1 \): In the next lemma, \( Y_{w_k} \) denotes the space of \( w_k \times w_k \) diagonal matrices for \( k \in [d-1] \). As before we assume \( w_0 = w_d = 1 \).

Lemma 3.4 (Diagonal Space) The space \( W_1 \) contains the space \( D_1 \oplus D_2 \oplus \cdots \oplus D_{d-1} \) such that for every \( k \in [d-1] \), \( D_k \) is isomorphic to the \( \mathbb{F} \)-linear space spanned by \( t_k \times t_k \) matrices of the form

\[
\begin{bmatrix}
-Y \otimes I_{w_{k-1}} & 0 \\
0 & I_{w_{k+1}} \otimes Y
\end{bmatrix}
t_k \times t_k
\]

where \( Y \in Y_{w_k} \) and \( t_k = w_k(w_{k-1} + w_{k+1}) \).

Hence, \( \dim(W_1) \geq \sum_{k=1}^{d-1} w_k \).
The proof is presented after we elaborate on the structure of matrices in \( \mathcal{D}_1 \).

**Elaboration on Lemma 3.4:** An element \( D \in \mathcal{D}_1 \oplus \mathcal{D}_2 \oplus \cdots \oplus \mathcal{D}_{d-1} \) is a sum of \( d - 1, n \times n \) matrices \( D_1, D_2, \ldots, D_{d-1} \) such that for every \( k \in [d - 1] \), \( D_k \in \mathcal{D}_k \) and the non-zero entries of \( D_k \) are restricted to the rows and columns indexed by \( x_k \cup x_{k+1} \) variables. The submatrix in \( D_k \) corresponding to these rows and columns looks as shown in Equation (3.11).

**Proof of Lemma 3.4:** The proof is similar to the proof of Lemma 3.3. Recall \( w_0 = w_d = 1 \) and \( \mathcal{Y}_{w_k} \) denotes the space of \( w_k \times w_k \) diagonal matrices. Every \( D \in \mathcal{W}_1 \) satisfies an equation of the following form

\[
\sum_{i \in [w_1]} d_i^{(1)} \frac{\partial \text{IMM}}{\partial x_i^{(1)}} + \sum_{k=2}^{d-1} \sum_{i \in [w_{k-1}], j \in [w_k]} d_{ij}^{(k)} x_{ij}^{(k)} \frac{\partial \text{IMM}}{\partial x_{ij}^{(k)}} + \sum_{i \in [w_{d-1}]} d_i^{(d)} x_i^{(d)} \frac{\partial \text{IMM}}{\partial x_i^{(d)}} = 0.
\]

A succinct way to write the above equation is

\[
\sum_{k=1}^{d} Q_1 Q_2 \cdots Q_{k-1} Q_k Q_{k+1} \cdots Q_d = 0,
\] (3.12)

where \( Q'_i = (d_i^{(1)} x_i^{(1)})_{i \in [w_1]} \) is a row vector, \( Q'_d = (d_i^{(d)} x_i^{(d)})_{i \in [w_{d-1}]} \) is a column vector, \( Q'_k = (d_{ij}^{(k)} x_{ij}^{(k)})_{i \in [w_{k-1}], j \in [w_k]} \) and \( Q_1, \ldots, Q_d \) are matrices as in Section 2.3. For every \( k \in [d - 1] \), let us focus on those \( D_k \in \mathcal{W}_1 \) for which the matrices \( Q'_1, \ldots, Q'_{k-1}, Q'_{k+2}, \ldots, Q'_d \) are zero in Equation (3.12). Such a \( D_k \) satisfies the following equation,

\[
Q_1 Q_2 \cdots Q_k Q_{k+1} \cdots Q_d + Q_1 Q_2 \cdots Q_k Q'_{k+1} \cdots Q_d = 0.
\] (3.13)

Using a similar argument as in the proof of Lemma 3.3 we get \( Q'_{k+1} = Y \cdot Q_k \) and \( Q'_k = -Q_k \cdot Y \) where \( Y \in \mathcal{Y}_{w_k} \). Further, any \( Y \in \mathcal{Y}_{w_k} \) can be used along with the relations \( Q'_1 = Y \cdot Q_k + 1 \) and \( Q'_k = -Q_k \cdot Y \) to satisfy Equation (3.13). The set of \( D_k \in \mathcal{W}_1 \) satisfying Equation (3.13) forms an \( \mathbb{F} \)-linear space; call it \( \mathcal{D}_k \). Every \( D_k \in \mathcal{D}_k \) is such that the submatrix defined by the rows and the columns labelled by the variables in \( x_k \) and \( x_{k+1} \) looks like

\[
\begin{bmatrix}
-Y \otimes I_{w_{k-1}} & 0 \\
0 & I_{w_{k+1}} \otimes Y
\end{bmatrix}_{t_k \times t_k}
\]

where \( Y \in \mathcal{Y}_{w_k} \) and \( t_k = w_k(w_{k-1} + w_{k+1}) \),

and all other entries in \( D_k \) are zero. Moreover, any \( n \times n \) matrix with this structure is in \( \mathcal{D}_k \). Thus \( \mathcal{D}_k \) is isomorphic to the space of all \( t_k \times t_k \) matrices of the form shown above. It can also

58
be easily verified that every matrix in \( D_1 + \ldots + D_{d-1} \) can be expressed uniquely as a sum of matrices in these spaces. Hence \( W_1 \supseteq D_1 \oplus D_2 \oplus \cdots \oplus D_{d-1} \) completing the proof of Lemma 3.4.

### 3.1.2 Random elements of \( g_{\text{IMM}} \)

In Lemma 3.5 we show that the characteristic polynomial of a random matrix \( R \) in \( g_{\text{IMM}} \) is square-free with high probability. From Claim 2.1 this implies that the characteristic polynomial of a random matrix \( R' \) in the Lie algebra of a polynomial \( f \) equivalent to IMM is also square-free with high probability. This is helpful in designing an equivalence testing algorithm for IMM (see Theorem 1.1a).

**Claim 3.4** There is a diagonal matrix \( D \in g_{\text{IMM}} \) with all entries distinct.

**Proof:** From Lemma 3.4, we know that for \( k \in [d-1] \) the submatrix of \( D_k \in D_k \) defined by the rows and columns indexed by the variables in \( x_k \uplus x_{k+1} \) is

\[
\begin{bmatrix}
-Y_k \otimes I_{w_k-1} & 0 \\
0 & I_{w_{k+1}} \otimes Y_k
\end{bmatrix},
\]

where \( Y_k \in Y_k \). Let the \((i,i)\)-th entry of \( Y_k \) be \( y_{i}^{(k)} \) and pretend that these entries are distinct formal variables, say \( y \) variables. Consider the matrix \( D = \sum_{i=1}^{d-1} D_i \) and observe the following:

a. For \( k \in [2, d-1] \), the \((x_{ij}^{(k)}, x_{ij}^{(k)})\)-th entry of \( D \) is \( y_{i}^{(k-1)} - y_{j}^{(k)} \) where \( i \in [w_{k-1}] \) and \( j \in [w_k] \).

b. The \((x_{i1}^{(1)}, x_{i1}^{(1)})\)-th and \((x_{j1}^{(d)}, x_{j1}^{(d)})\)-th entry of \( D \) are \(-y_{i}^{(1)} \) and \( y_{j}^{(d-1)} \) respectively, where \( i \in [w_1] \) and \( j \in [w_{d-1}] \).

In particular, all the diagonal entries of \( D \) are distinct linear forms in the \( y \) variables. Hence, if we assign values to the \( y \) variables uniformly at random from a set \( S \subseteq F \) such that \(|S| \geq n^2\) then with non zero probability \( D \) has all diagonal entries distinct after the random assignment.

**Lemma 3.5** If \( \{L_1, L_2, \ldots, L_m\} \) is a basis of the Lie algebra \( g_{\text{IMM}} \) then the characteristic polynomial of an element \( L = \sum_{i=1}^{m} r_i L_i \), where \( r_i \in R \) \( F \) is picked independently and uniformly at random from \( [2n^3] \), is square free with probability at least \( 1 - \frac{1}{\mu^{O(1)}} \).
Observation 3.3 The discriminant of $h_r(x)$, $\text{disc}(h_r(x)) := \text{res}_x(h_r, \frac{\partial h_r}{\partial x})$, is a non zero polynomial in $r$ variables of degree at most $1 2n^2$, where $\text{res}_x(h_r, \frac{\partial h_r}{\partial x})$ is the resultant of $h_r$ and $\frac{\partial h_r}{\partial x}$ when treated as univariates in $x$.

Observation 3.3 is proved at the end of the section. Since $\text{disc}(h_r(x))$ is not an identically zero polynomial in the $r$ variables and has degree less than $2n^2$, if we set every $r$ variable uniformly and independently at random to a value in $[2n^2]$ then using Schwartz-Zippel lemma with probability at least $1 - \frac{1}{n^{\omega(1)}}$, $\gcd(h_r, \frac{\partial h_r}{\partial x}) = 1$. This implies with probability at least $1 - \frac{1}{n^{\omega(1)}}$, $h_r(x)$ is square free.

Proof of Observation 3.3: $h_r$ is a monic polynomial in $x$ of degree $n$ and $\frac{\partial h_r}{\partial x}$ is a polynomial in $x$ of degree $(n - 1)$. Also the coefficient of $x^{n-1}$ in $\frac{\partial h_r}{\partial x}$ is $r$ variable free. The Sylvester matrix of $h_r$ and $\frac{\partial h_r}{\partial x}$ with respect to variable $x$ is a $(2n - 1) \times (2n - 1)$ matrix. Thus, $\text{res}_x(h_r, \frac{\partial h_r}{\partial x})$ is a polynomial in the $r$-variables of degree less than $2n^2$. If $\text{res}_x(h_r, \frac{\partial h_r}{\partial x})$ is identically zero as a polynomial in $r$ then for every setting of $r$ to field elements $\gcd(h_r, \frac{\partial h_r}{\partial x}) \neq 1$ implying $h_r$ is not square free. This would contradict Claim 3.4 as we can set the $r$ variables appropriately such that $L$ is a diagonal matrix with distinct diagonal entries, and $h_r$ for such a setting of the $r$ variables is square free.

3.1.3 Invariant subspaces of $\mathfrak{g}_{\text{IMM}}$

The ordering of the variables in IMM allows us to identify them naturally with the unit vectors $e_1, e_2, \ldots, e_n$ in $\mathbb{R}^n$ – the vector $e_i$ corresponds to the $i$-th variable in the ordering. We will write $e_x$ to refer to the unit vector corresponding to the variable $x$. Let $\mathcal{U}_{1,2}$ represent the coordinate subspace spanned by the unit vectors corresponding to the variables in $x_1 \uplus x_2$. Similarly $\mathcal{U}_k$ represents the coordinate subspace spanned by the unit vectors corresponding to the variables in $x_k$ for $k \in [2, d - 1]$, and $\mathcal{U}_{d-1,d}$ represents the coordinate subspace spanned by the unit vectors corresponding to the variables in $x_{d-1} \uplus x_d$. In Lemma 3.6, we establish that $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$ are the only irreducible invariant subspaces of $\mathfrak{g}_{\text{IMM}}$.

Claim 3.5 Let $\mathcal{U}$ be a nonzero invariant subspace of $\mathfrak{g}_{\text{IMM}}$. If $u = (u_1, u_2, \ldots, u_n)^T \in \mathcal{U}$ and $u_j \neq 0$ then $e_j \in \mathcal{U}$, implying $\mathcal{U}$ is a coordinate subspace.

1A careful analysis could show that the degree is in fact $n(n - 1)$, but we do not need such a precision here.
\textbf{Proof:} Claim 3.4 states that there is a diagonal matrix \( D \in \mathfrak{g}_{\text{hom}} \) with distinct diagonal entries \( \lambda_1, \lambda_2, \ldots, \lambda_n \). Since \( \mathcal{U} \) is invariant for \( D \), if \( u = (u_1, u_2, \ldots, u_n)^T \in \mathcal{U} \) then \( (\lambda_1^i u_1, \lambda_2^i u_2, \ldots, \lambda_n^i u_n) \in \mathcal{U} \) for every \( i \in \mathbb{N} \). Let \( S_u := \{ j \in [n] \mid u_j \neq 0 \} \) be the support of \( u \). As \( \lambda_1, \lambda_2, \ldots, \lambda_n \) are distinct, the vectors \( (\lambda_1^i u_1, \lambda_2^i u_2, \ldots, \lambda_n^i u_n) \) are \( \mathbb{F} \)-linearly independent for \( 0 \leq i < |S_u| \). Hence, the unit vector \( e_j \in \mathcal{U} \) for every \( j \in S_u \). It follows that \( \mathcal{U} \) is the coordinate subspace spanned by those \( e_j \) for which \( j \in S_u \) for some \( u \in \mathcal{U} \).

**Lemma 3.6** The only irreducible invariant subspaces of \( \mathfrak{g}_{\text{hom}} \) are \( \mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d} \).

\textbf{Proof:} It follows from Lemma 3.1 and Figure 3.1 that \( \mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d} \) are invariant subspaces. We show in the next two claims that the spaces \( \mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d} \) are irreducible.

**Claim 3.6** No invariant subspace of \( \mathfrak{g}_{\text{hom}} \) is properly contained in \( \mathcal{U}_k \) for \( k \in [2, d - 1] \).

\textbf{Proof:} Let \( \mathcal{U} \subseteq \mathcal{U}_k \) be an invariant subspace of \( \mathfrak{g}_{\text{hom}} \). From Claim 3.5 it follows that \( \mathcal{U} \) is a coordinate subspace. For \( t \in \mathbb{N} \), let \( \tilde{I}_t \stackrel{\text{def}}{=} I_t - I_s \), where \( I_t \) is the \( t \times t \) all one matrix. From Lemma 3.3, there are matrices \( B_{k-1} \) and \( B_k \) in \( \mathfrak{g}_{\text{hom}} \) such that the submatrix of \( B_{k-1} \) restricted to the rows and the columns labelled by the variables in \( x_{k-1} \cup x_k \) looks like

\[
\begin{bmatrix}
-\tilde{I}_{w_{k-1}} \otimes I_{w_{k-2}} & 0 \\
0 & I_{w_k} \otimes \tilde{I}_{w_{k-1}}
\end{bmatrix}, \text{ and}
\]

the submatrix in \( B_k \) restricted to the rows and the columns labelled by the variables in \( x_k \cup x_{k+1} \) looks like

\[
\begin{bmatrix}
\tilde{I}_{w_k} \otimes I_{w_{k-1}} & 0 \\
0 & I_{w_{k+1}} \otimes -\tilde{I}_{w_k}
\end{bmatrix}.
\]

From Lemma 3.4, there is a diagonal matrix \( D_{k-1} \) in \( \mathfrak{g}_{\text{hom}} \) such that the submatrix restricted to the rows and the columns labelled by the variables in \( x_{k-1} \cup x_k \) looks like

\[
\begin{bmatrix}
-I_{w_{k-1}} \otimes I_{w_{k-2}} & 0 \\
0 & I_{w_k} \otimes I_{w_{k-1}}
\end{bmatrix}.
\]

Let \( L = B_{k-1} + B_k + D_{k-1} \). The submatrix of \( L \) restricted to the rows and the columns labelled by the variables in \( x_k \) looks as shown in Figure 3.7. For notational simplicity we write \( w_{k-1} \) as \( w \) in Figure 3.7. If \( e_x \) is a unit vector in \( \mathcal{U} \), where \( x = x^{(k)}_{ij} \) is a variable in \( x_k \) then the matrix \( L \) maps \( e_x \) to \( Le_x \) which is the column of \( L \) labelled by the variable \( x \). This column vector has all entries zero except for the rows labelled by the variables in \( x_k \). Restricting to
Figure 3.7: Submatrix of $L$ restricted to rows/columns indexed by $x_k$

de these rows and looking at Figure 3.7, we infer that the rows of $Le_x$ labelled by the variables $x_{1j}^{(k)}, x_{2j}^{(k)}, \ldots, x_{w_{k-1}j}^{(k)}$ are 1 (in particular, these entries are nonzero). We use this knowledge and that $Le_x \in \mathcal{U}$ to make the following observation, the proof of which is immediate from Claim 3.5.

Observation 3.4 If $e_x \in \mathcal{U}$, where $x = x_{ij}^{(k)}$ then $e_{x'} \in \mathcal{U}$ for every $x' \in \{x_{1j}^{(k)}, x_{2j}^{(k)}, \ldots, x_{w_{k-1}j}^{(k)}\}$.

Moreover, it follows from the presence of $I_w$ matrices in Figure 3.7 that for every $j' \in [w_k]$ there is the variable $y = x_{ij'}^{(k)}$ such that the row labelled by $y$ in $Le_x$ is 1, implying $e_y \in \mathcal{U}$. Hence from Observation 3.4, $e_{y'} \in \mathcal{U}$ for every $y' \in \{x_{ij'}^{(k)}, \ldots, x_{w_{k-1}j'}^{(k)}\}$. Since this is true for every $j' \in [w_k]$, $e_y \in \mathcal{U}$ for every variable $y \in x_k$ implying $\mathcal{U} = \mathcal{U}_k$.

Claim 3.7 The invariant subspaces $\mathcal{U}_{1,2}$ and $\mathcal{U}_{d-1,d}$ are irreducible, and the only invariant subspace properly contained in $\mathcal{U}_{1,2}$ (respectively $\mathcal{U}_{d-1,d}$) is $\mathcal{U}_2$ (respectively $\mathcal{U}_{d-1}$).

Proof: We prove the claim for $\mathcal{U}_{1,2}$, the proof for $\mathcal{U}_{d-1,d}$ is similar. Suppose $\mathcal{U}_{1,2} = \mathcal{V} \oplus \mathcal{W}$ where $\mathcal{V}, \mathcal{W}$ are invariant subspaces of $\mathfrak{g}_{\text{imm}}$ (and so also coordinate subspaces). A unit vector $e_x$, where $x \in x_1$ is either in $\mathcal{V}$ or $\mathcal{W}$. Suppose $e_x \in \mathcal{V}$; we will show that $\mathcal{V} = \mathcal{U}_{1,2}$.

1follows again from Claim 3.5.
matrix $M \in \mathfrak{g}_{\text{bmm}}$ such that the submatrix of $M$ restricted to the rows and the columns labelled by the variables in $x_1$ and $x_2$ looks as shown in Figure 3.8, in which $w = w_1$ and $C$ is a $w \times w$ anti-symmetric matrix with all non-diagonal entries nonzero. All the other entries of $M$ are zero. The vector $Me_x$ is the first column of $M$ and it is zero everywhere except for the rows

![Figure 3.8: Submatrix of $M$ matrix restricted to rows/columns indexed by $x_1 \uplus x_2$](image)

labelled by the variables in $x_1 \uplus x_2$. Among these rows, unless $y \in \{x_{11}^{(2)}, x_{12}^{(2)}, \ldots, x_{1w_2}^{(2)}\}$ the row of $Me_x$ labelled by $y$ is nonzero. Thus (from Claim 3.5), $e_y \in \mathcal{V}$ for $y \in x_1$ and $y = x_{ij}^{(2)}$ where $i \in [2, w_1]$ and $j \in [w_2]$. Let $y = x_{ij}^{(2)}$ for some $i \in [2, w_1]$ and $j \in [w_2]$. From Figure 3.8, the row of $Me_y$ labelled by $x_{1j}^{(2)}$ is nonzero and so, for $y' = x_{ij}^{(2)}$, $e_{y'}$ is also in $\mathcal{V}$. Hence, $\mathcal{V} = \mathcal{U}_{1,2}$ and $\mathcal{U}_{1,2}$ is irreducible. To argue that the only invariant subspace properly contained in $\mathcal{U}_{1,2}$ is $\mathcal{U}_2$, let $\mathcal{V} \subset \mathcal{U}_{1,2}$ be an invariant subspace of $\mathfrak{g}_{\text{bmm}}$. From the above argument it follows that $e_x \notin \mathcal{V}$ for every $x \in x_1$ (otherwise $\mathcal{V} = \mathcal{U}_{1,2}$). This implies $\mathcal{V} \subseteq \mathcal{U}_2$ and from Claim 3.6 we have $\mathcal{V} = \mathcal{U}_2$.

In the proof of Claim 3.6 we have shown that the closure of $e_x$ under the action of $\mathfrak{g}_{\text{bmm}}$ is $\mathcal{U}_k$ for any $x \in x_k$, where $k \in [2, d - 1]$. Similarly, in the proof of Claim 3.7 we have shown that the closure of $e_x$ under the action of $\mathfrak{g}_{\text{bmm}}$ is $\mathcal{U}_{1,2}$ (respectively $\mathcal{U}_{d-1,d}$) for any $x \in x_1$ (respectively $x \in x_d$). This observation helps infer that the spaces $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$ are the only irreducible invariant subspaces of $\mathfrak{g}_{\text{bmm}}$. Suppose $\mathcal{V}$ is an irreducible invariant subspace. If $e_x \in \mathcal{V}$ for some $x \in x_k$ where $k \in [2, d - 1]$, then $\mathcal{U}_k \subseteq \mathcal{V}$ as $\mathcal{U}_k$ is the closure of $e_x$. If $e_x \in \mathcal{V}$ for some $x \in x_1$ (respectively $x \in x_d$) then $\mathcal{U}_{1,2} \subseteq \mathcal{V}$ (respectively $\mathcal{U}_{d-1,d} \subseteq \mathcal{V}$) as $\mathcal{U}_{1,2}$ (respectively $\mathcal{U}_{d-1,d}$) is the closure of $e_x$. Therefore $\mathcal{V}$ is a direct sum of some of the irreducible invariant subspaces $\mathcal{U}_{1,2}, \mathcal{U}_2, \ldots, \mathcal{U}_{d-1}, \mathcal{U}_{d-1,d}$. Since $\mathcal{V}$ is irreducible, it is equal to 63
one of these irreducible invariant subspaces.

**Corollary 3.1 (Uniqueness of decomposition)** The decomposition,

\[ \mathbb{F}^n = U_{1,2} \oplus U_3 \oplus \cdots \oplus U_{d-2} \oplus U_{d-1,d} \]

is unique in the following sense; if \( \mathbb{F}^n = V_1 \oplus V_2 \oplus \cdots \oplus V_s \), where \( V_i \)'s are irreducible invariant subspaces of \( g_{mm} \), then \( s = d - 2 \) and for every \( i \in [s] \), \( V_i \) is equal to \( U_{1,2} \) or \( U_{d-1,d} \) or some \( U_k \) for \( k \in [3, d - 2] \).

**Proof:** Since \( V_i \)'s are irreducible invariant subspaces, from Lemma 3.6 it follows that for every \( i \in [s] \) \( V_i \) equals one among \( U_{1,2} \), \( U_2 \), \( \ldots \), \( U_{d-1,d} \). Since \( V_1, V_2, \ldots, V_s \) span the entire \( \mathbb{F}^n \), the only possible decomposition is \( \mathbb{F}^n = U_{1,2} \oplus U_3 \oplus \cdots \oplus U_{d-2} \oplus U_{d-1,d} \). □

### 3.2 Lie algebra of \( \text{Det}_n \)

Before delving into the Lie algebra of the determinant polynomial we set up some notations that are useful in the arguments presented in this section.

**Notations:** Let \( X = (x_{i,j})_{i,j \in [n]} \) and \( \text{Det}_n \) be the determinant of \( X \). We reuse symbols to denote the set of variables in \( \text{Det}_n \) as \( x = \{x_{1,1}, x_{1,2}, \ldots, x_{n,n}\} \), and the define an ordering among these variables as follows: Order the \( x \) variables in a column-major fashion, and within a column \( i \in [n] \) the variables are ordered as \( x_{1,i} > x_{2,i} > \ldots > x_{n,i} \). We will prove the following theorem. We use \( \mathbb{Z}_n \) to denote the space of \( n \times n \) traceless matrices, and \( L_{\text{row}}, L_{\text{col}} \) to denote the spaces of \( n^2 \times n^2 \) matrices \( \mathbb{Z}_n \otimes I_n, I_n \otimes \mathbb{Z}_n \) respectively. Lastly, we assume that the the rows and columns of the \( n^2 \times n^2 \) matrices in \( g_{\text{Det}_n} \) are indexed by the variables of \( \text{Det}_n \) ordered as above.

This section is devoted to proving Lemma 3.7 which describes the structure of matrices in the Lie algebra of \( \text{Det}_n \). Lemma 3.7 is well-known, and we present a proof here for completeness.

**Lemma 3.7** Let \( n \in \mathbb{N}^\times \) and \( \mathbb{F} \) a field such that \( \text{char}(\mathbb{F}) \nmid n \). Then the Lie algebra of \( \text{Det}_n \) is equal to the direct sum of two sub Lie algebras \( L_{\text{row}} \) and \( L_{\text{col}} \), i.e. \( g_{\text{Det}_n} = L_{\text{row}} \oplus L_{\text{col}} \).

**Proof:** Since \( L_{\text{row}} \cap L_{\text{col}} = \{0\} \), it is sufficient to show \( g_{\text{Det}_n} = L_{\text{row}} + L_{\text{col}} \). An element \( E \in g_{\text{Det}_n} \) satisfies

\[
\sum_{i_1,j_1,i_2,j_2 \in [n]} e_{(i_1,j_1),(i_2,j_2)} x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}} = 0 .
\]

(3.14)

Here \( e_{(i_1,j_1),(i_2,j_2)} \) is the entry of \( E \) whose row and column are indexed by \( x_{i_1,j_1} \) and \( x_{i_2,j_2} \) respectively. For convenience, if \( i_1 = i_2 \) and \( j_1 = j_2 \) then we denote \( e_{(i_1,j_1),(i_1,j_1)} \) as \( e_{i_1,j_1} \). We will
show that $E$ satisfies Equation 3.14 if and only if the entries of $E$ satisfy a set of equations: Equation 3.18a, 3.18b, 3.18c, and 3.19. In particular these equations together characterize the matrices in $g_{\text{Det}_n}$.

We focus on a term $x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}}$ and observe the following.

**Observation 3.5** Any monomial of $x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}}$ is a monomial of $\text{Det}_n$ if and only if $i_1 = i_2$ and $j_1 = j_2$. (i.e we derive and shift by the same variable)

From Observation 3.5 we can split Equation 3.14 as follows.

$$
\sum_{i_1,j_1 \in [n]} \sum_{i_1 \neq i_2 \text{ or } j_1 \neq j_2} \sum_{i_1,j_1 \in [n]} e_{(i_1,j_1)(i_2,j_2)} x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}} = 0.
$$

(3.15a)

$$
\sum_{i_1,j_1 \in [n]} e_{i_1,j_1} x_{i_1,j_1} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}} = 0.
$$

(3.15b)

Equation 3.15a corresponds to the non-diagonal entries of $E$ and Equation 3.15b corresponds to the diagonal entries of $E$. We analyze these two sets of entries separately. First let us focus on the non-diagonal entries.

**Non-diagonal entries:** With respect to the entries of $E$ in Equation 3.15a we show the following.

**Claim 3.8** In Equation 3.15a, if $i_1 \neq i_2$ and $j_1 \neq j_2$ then $e_{(i_1,j_1)(i_2,j_2)} = 0$.

**Proof:** View the polynomial $x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}}$ as a univariate polynomial in $x_{i_2,j_2}$ over the ring $\mathbb{F}[x \setminus \{x_{i_2,j_2}\}]$. Since $i_1 \neq i_2$ and $j_1 \neq j_2$, degree of $x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}}$ is equal to two. Let $x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}} = x_{i_2,j_2}^2 \cdot f + x_{i_2,j_2} \cdot g$, where $f, g \in \mathbb{F}[x \setminus \{x_{i_2,j_2}\}]$. The claim follows by observing that the monomials in $x_{i_2,j_2}^2 \cdot f$ are unique to the term $x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}}$, that is any monomial in $x_{i_2,j_2}^2 \cdot f$ is a monomial in $x_{p_2,q_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{p_1,q_1}}$ if and only if $x_{i_1,j_1} = x_{p_1,q_1}$ and $x_{i_2,j_2} = x_{p_2,q_2}$.

From Claim 3.8, we have that the non-zero terms in Equation 3.15a correspond to terms where we derive and shift by variables in the same row, or where we derive and shift by variables in the same column. Thus, we may rewrite Equation 3.15a as follows.

$$
\sum_{i,j_2 \in [n]} e_{(i_1,j_1)(i_2,j_2)} x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}} + \sum_{i_1,j_2 \in [n]} e_{(i_1,j_1)(i_2,j_2)} x_{i_2,j_2} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1,j_1}} = 0.
$$

(3.16)
Focus on the terms $x_{i,j_2} \cdot \frac{\partial \text{det}_n}{\partial x_{i,j_1}}$ and $x_{i_2,j} \cdot \frac{\partial \text{det}_n}{\partial x_{i_1,j}}$ and observe the following.

**Observation 3.6**  
1. For $i, j_1, j_2 \in [n]$ and $j_1 \neq j_2$, any monomial in $x_{i,j_2} \cdot \frac{\partial \text{det}_n}{\partial x_{i,j_1}}$ has one variable from each row, no variable from column $j_1$, two variables from column $j_2$, and one variable from every other column apart from $j_1$ and $j_2$.

2. For $i_1, i_2, j \in [n]$ and $i_1 \neq i_2$, any monomial in $x_{i_2,j} \cdot \frac{\partial \text{det}_n}{\partial x_{i_1,j}}$ has one variable from every column, no variable from row $i_1$, two variables from row $i_2$, and one variable from every other row apart from $i_1$ and $i_2$.

In particular terms $\sum_{i,j_1,j_2 \in [n]} e_{(i,j_1),(i,j_2)} x_{i,j_2} \frac{\partial \text{det}_n}{\partial x_{i,j_1}}$ and $\sum_{i_1,i_2,j \in [n]} e_{(i_1,j),(i_2,j)} x_{i_2,j} \frac{\partial \text{det}_n}{\partial x_{i_1,j}}$ are monomial disjoint, and hence

\[
\sum_{i,j_1,j_2 \in [n]} e_{(i,j_1),(i,j_2)} x_{i,j_2} \frac{\partial \text{det}_n}{\partial x_{i,j_1}} = 0. 
\tag{3.17a}
\]

\[
\sum_{i_1,i_2,j \in [n]} e_{(i_1,j),(i_2,j)} x_{i_2,j} \frac{\partial \text{det}_n}{\partial x_{i_1,j}} = 0. 
\tag{3.17b}
\]

The claim below helps us to characterize the non-diagonal entries of $E$.

**Claim 3.9**  
1. Let $\mu$ be a monomial of the term $x_{i,j_2} \cdot \frac{\partial \text{det}_n}{\partial x_{i,j_1}}$ in Equation 3.17a such that the two variables of column $j_2$ are $x_{i,j_2}$ and $x_{p,j_2}$ for $p \in [n]$ and $p \neq i$. Then $\mu$ is a monomial of the term $x_{p_1,q_2} \cdot \frac{\partial \text{det}_n}{\partial x_{p_1,q_1}}$ in Equation 3.17a if and only if $p_1 = i$ or $p_1 = p$, and $q_2 = j_2$ and $q_1 = j_1$. Further, if $\text{Char}(F) \neq 2$ then the coefficient of $\mu$ in $x_{i,j_2} \cdot \frac{\partial \text{det}_n}{\partial x_{i,j_1}}$ and $x_{p,j_2} \cdot \frac{\partial \text{det}_n}{\partial x_{p,j_1}}$ are 1 and -1, or -1 and 1 respectively (i.e they have opposite signs).

2. Let $\mu$ be a monomial of the term $x_{i_2,j} \cdot \frac{\partial \text{det}_n}{\partial x_{i_1,j}}$ in Equation 3.17b such that the two variables of row $i_2$ are $x_{i_2,j}$ and $x_{i_2,q}$ for $q \in [n]$ and $q \neq j$. Then $\mu$ is a monomial of the term $x_{p_2,q_1} \cdot \frac{\partial \text{det}_n}{\partial x_{p_1,q_1}}$ in Equation 3.17b if and only if $p_1 = i_1$ and $p_2 = i_2$, and $q_1 = j$ or $q_1 = q$. Further, if $\text{Char}(F) \neq 2$ then the coefficient of $\mu$ in $x_{i_2,j} \cdot \frac{\partial \text{det}_n}{\partial x_{i_1,j}}$ and $x_{i_2,q} \cdot \frac{\partial \text{det}_n}{\partial x_{i_1,q}}$ are 1 and -1, or -1 and 1 respectively (i.e they have opposite signs).

**Proof:** We prove part 1 of the claim, the proof of part 2 is similar. Since $\mu$ is a monomial in $x_{i,j_2} \cdot \frac{\partial \text{det}_n}{\partial x_{i,j_1}}$, $\mu$ has no variable from column $j_1$ and two variables from column $j_2$ and one variable from every other column apart from $j_1$ and $j_2$ (from Observation 3.6). Since $q_1 \neq q_2$ in Equation 3.17a it must be that $q_1 = j_1$ and $q_2 = j_2$. Further, as $x_{p_1,j_2}$ is part of every monomial in $x_{p_1,j_2} \cdot \frac{\partial \text{det}_n}{\partial x_{p_1,j_1}}$, we have $p_1 = i$ or $p_1 = p$. 

66
We now prove that the signs of the coefficients of $\mu$ in the two terms are opposite. Let $\mu_1 = \frac{\mu x_{i,j}}{x_{i,j}}$ and $\mu_2 = \frac{\mu x_{p,j}}{x_{p,j}}$. Then observe that the monomials $\mu_1$ and $\mu_2$ are monomials of $\text{Det}_n$, and the coefficient of $\mu$ in $x_{i,j} \cdot \frac{\partial \text{Det}_n}{\partial x_{i,j}}$ and $x_{p,j} \cdot \frac{\partial \text{Det}_n}{\partial x_{p,j}}$ are the coefficients of $\mu_1$ and $\mu_2$ in $\text{Det}_n$. Since $\mu_1$ and $\mu_2$ are monomials of $\text{Det}_n$, there are two permutations $\sigma, \tau$ of $[n]$, such that $\mu_1 = \prod_{k=1}^n x_{k, \sigma(k)}$, and $\mu_2 = \prod_{k=1}^n x_{k, \tau(k)}$ and the coefficient of $\mu_1, \mu_2$ in $\text{Det}_n$ are the signs of the permutation $\sigma, \tau$ respectively. From the definition of $\mu_1$ and $\mu_2$, we have for all $k \in [n]$, $k \neq i$, and $k \neq p$ $\sigma(k) = \tau(k)$. Also $\sigma(i) = j_1$, $\sigma(p) = j_2$, $\tau(i) = j_2$, and $\tau(p) = j_1$. Hence $\tau = (j_1, j_2) \cdot \sigma$, where $(j_1, j_2)$ denotes the transposition that swaps $j_1$ and $j_2$. This implies the signs of $\sigma$ and $\tau$ are opposite of each other.

It follows from Claims 3.8 and 3.9 that the non-diagonal elements of a matrix $E$ in $\mathfrak{g}_{\text{Det}_n}$ are characterized by Equations 3.18a, 3.18b and 3.18c.

\begin{align*}
e_{(i_1, j_1), (i_2, j_2)} &= 0 & \text{for all } i_1, i_2, j_1, j_2 \in [n], i_1 \neq i_2 \text{ and } j_1 \neq j_2 . & (3.18a) \\
e_{(i_1, j_1), (i_2, j_2)} &= e_{(i_2, j_1), (i_1, j_2)} & \text{for all } i_1, i_2, j_1, j_2 \in [n] \text{ and } j_1 \neq j_2 . & (3.18b) \\
e_{(i_1, j_1), (i_2, j_1)} &= e_{(i_1, j_2), (i_2, j_2)} & \text{for all } i_1, i_2, j_1, j_2 \in [n] \text{ and } i_1 \neq i_2 . & (3.18c)
\end{align*}

The matrix below depicts the structure of the non-diagonal entries of a matrix $E$ that satisfies Equations 3.18a, 3.18b, and 3.18c, when $n = 2$.

\[
\begin{bmatrix}
* & e_{(1,1), (1,2)} & e_{(1,1), (2,1)} & 0 \\
e_{(1,2), (1,1)} & * & 0 & e_{(1,1), (2,1)} \\
e_{(2,1), (1,1)} & 0 & * & e_{(1,1), (2,1)} \\
0 & e_{(2,1), (1,1)} & e_{(2,1), (1,1)} & *
\end{bmatrix}
\]

The diagonal elements of the above matrix are denoted by $\ast$. Notice $e_{(2,1), (2,2)} = e_{(1,1), (1,2)}$, and $e_{(2,2), (2,1)} = e_{(1,2), (1,1)}$ due to Equation 3.18b and, $e_{(1,2), (2,2)} = e_{(1,1), (2,1)}$ and $e_{(2,2), (1,2)} = e_{(2,1), (1,1)}$ due to Equation 3.18c. Next we analyse the diagonal elements of $E$ by analysing the terms in Equation 3.15b.

**Diagonal elements:** Observe that for a monomial $\mu = x_{i, \sigma(i)}$ in $\text{Det}_n$, where $\sigma$ is a permutation of $[n]$, $\mu$ is a monomial in term $x_{i_1, j_1} \cdot \frac{\partial \text{Det}_n}{\partial x_{i_1, j_1}}$ of Equation 3.15b, if and only if $j_1 = \sigma(i_1)$.
Thus the diagonal elements of a matrix $E$ in $\mathfrak{g}_{\text{Det}_n}$ are characterized by Equation 3.19.

\[ \sum_{i\in[n]} e_{i,\sigma(i)} = 0 \quad \text{for all permutations } \sigma \text{ of } [n]. \]  

(3.19)

From the above discussion it is easy to make the observation below.

**Observation 3.7** A matrix $E \in \mathfrak{g}_{\text{Det}_n}$ if and only if the entries of $E$ satisfy Equations 3.18a, 3.18b, 3.18c and 3.19.

Further it is easy to observe that a matrix in $\mathcal{L}_{\text{row}} \oplus \mathcal{L}_{\text{col}}$ satisfies Equations 3.18a, 3.18b, 3.18c and 3.19, and hence from the above Observation $\mathcal{L}_{\text{row}} + \mathcal{L}_{\text{col}} \subseteq \mathfrak{g}_{\text{Det}_n}$. Claim 3.10 shows that $\mathfrak{g}_{\text{Det}_n} \subseteq \mathcal{L}_{\text{row}} + \mathcal{L}_{\text{col}}$.

**Claim 3.10** Let $E \in \mathfrak{g}_{\text{Det}_n}$. Then there are matrices $M, N \in \mathbb{Z}_n$ such that $E = M \otimes I_n + I_n \otimes N$.

**Proof:** The matrices $M = (m_{i,j})_{i,j \in [n]}, N = (n_{i,j})_{i,j \in [n]}$ are defined as following. For $i, j \in [n]$ and $i \neq j$, $m_{i,j} = e_{(i,1),(j,1)}$ and $n_{i,j} = e_{(1,i),(1,j)}$. For $i \in [n]$, let $a_i = \sum_{j \in [n]} e_{i,j} 1$, and for $i, j \in [n]$ let $e'_{i,j} = e_{i,j} - a_i$. Then for $i \in [n]$ $m(i,i) = a_i$, and $n(i,i) = e'_{i,i}$. The matrix below, depicts $M \otimes I_n + I_n \otimes N$, for $n = 3$. We will show $E = M \otimes I_n + I_n \otimes N$, and $M, N \in \mathbb{Z}_n$.

\[
\begin{bmatrix}
    a_1 + e'_{1,1} & n_{1,2} & n_{1,3} & m_{1,2} & 0 & 0 & m_{1,3} & 0 & 0 \\
    n_{2,1} & a_1 + e'_{1,2} & n_{2,3} & 0 & m_{1,2} & 0 & 0 & m_{1,3} & 0 \\
    n_{3,1} & n_{3,2} & a_1 + e'_{1,3} & 0 & 0 & m_{1,2} & 0 & 0 & m_{1,3} \\
    m_{2,1} & 0 & 0 & a_2 + e'_{1,1} & n_{1,2} & n_{1,3} & m_{2,3} & 0 & 0 \\
    0 & m_{2,1} & 0 & n_{2,1} & a_2 + e'_{1,2} & n_{2,3} & 0 & m_{2,3} & 0 \\
    0 & 0 & m_{2,1} & n_{3,1} & n_{3,2} & a_2 + e'_{1,3} & 0 & 0 & m_{2,3} \\
    m_{3,1} & 0 & 0 & m_{3,2} & 0 & 0 & a_3 + e'_{1,1} & n_{1,2} & n_{1,3} \\
    0 & m_{3,1} & 0 & 0 & m_{3,2} & 0 & n_{2,1} & a_3 + e'_{1,2} & n_{2,3} \\
    0 & 0 & m_{3,1} & 0 & 0 & m_{3,2} & n_{3,1} & n_{3,2} & a_3 + e'_{1,3}
\end{bmatrix}
\]

Since $E \in \mathfrak{g}_{\text{Det}_n}$, the non-diagonal entries of $E$ satisfy Equations 3.18a, 3.18b and 3.18c (from Observation 3.7). Hence the non-diagonal entries of $E$ are equal to the non-diagonal entries of $I_n \otimes M + N \otimes I_n$. Also $\sum_{i \in [n]} e'_{1,i} = 0$ implies $N \in \mathbb{Z}_n$. To show that the diagonal entries of these two matrices are equal, it is sufficient to show that for all $i, j \in [n], e'_{i,j} = e'_{i,j}$.

\footnote{Over finite fields, if \text{char}(\mathbb{F}) \nmid n then this is equivalent to multiplying by the multiplicative inverse of $n \cdot 1_\mathbb{F}$, where $1_\mathbb{F}$ is multiplicative identity of $\mathbb{F}$.}

68
We will show this using Equation 3.19, which is satisfied by the diagonal entries of $E$ (from Observation 3.7). Let $t = \sum_{i \in [n]} a_i$. Consider the following equations we get corresponding to different permutations.

\begin{align*}
e_{j,j} + \sum_{q \in [n], q \neq j} e_{q,q} &= e'_{j,j} + (\sum_{q \in [n], q \neq j} e'_{q,q}) + t = 0 \quad \text{(3.20a)} \\
e_{j,i} + e_{i,j} + \sum_{q \in [n], q \neq i, q \neq j} e_{q,q} &= e'_{j,i} + e'_{i,j} + (\sum_{q \in [n], q \neq i, q \neq j} e'_{q,q}) + t = 0 \quad \text{(3.20b)} \\
e_{j,j} + e_{p,i} + e_{i,p} + \sum_{q \in [n], q \neq j} e_{q,q} &= e'_{j,j} + e'_{p,i} + e'_{i,p} + (\sum_{q \in [n], q \neq j} e'_{q,q}) + t = 0 \quad p \in [n], p \neq i, p \neq j \quad \text{(3.20c)} \\
e_{p,i} + e_{i,j} + e_{j,p} + \sum_{q \in [n], q \neq j} e_{q,q} &= e'_{p,i} + e'_{i,j} + e'_{j,p} + (\sum_{q \in [n], q \neq j} e'_{q,q}) + t = 0 \quad p \in [n], p \neq i, p \neq j \quad \text{(3.20d)}
\end{align*}

Equation 3.20a corresponds to the identity permutation on $[n]$, Equation 3.20b corresponds to the transposition $\langle i, j \rangle$, Equation 3.20c corresponds to the transposition $\langle p, i \rangle$ and Equation 3.20d corresponds to the cycle $\langle p, i, j \rangle$. Subtracting Equation 3.20b from Equation 3.20a, we have

$$e'_{j,j} - e'_{j,i} = e'_{i,j} - e'_{i,i} \quad \text{(3.21)}$$

Similarly subtracting Equation 3.20d from Equation 3.20c, for all $p \in [n]$, and $p \neq i, p \neq j$ we have

$$e'_{j,j} - e'_{j,p} = e'_{i,j} - e'_{i,p} \quad \text{(3.22)}$$

Adding Equation 3.21 and Equations 3.22 for all $p \in [n]$, and $p \neq i, p \neq j$ we have

$$(n-1)e'_{j,j} - \sum_{p \in [n], p \neq j} e'_{j,p} = (n-1)e'_{i,j} - \sum_{p \in [n], p \neq j} e'_{i,p} \, .$$

We add and subtract $e'_{j,j}$ and $e'_{i,j}$ to the LHS and RHS of the above equation respectively,

$$n \cdot e'_{j,j} - \sum_{p \in [n]} e'_{j,p} = n \cdot e'_{i,j} - \sum_{p \in [n]} e'_{i,p} \, .$$

Since $\sum_{p \in [n]} e'_{j,p} = 0$, and $\sum_{p \in [n]} e'_{i,p} = 0$, and char$(F) \nmid n$ it follows that $e'_{j,j} = e'_{i,j}$. Since $e'_{j,j} = e'_{i,j}$ for all $i, j \in [n]$, from Equation 3.20a we have $t = \sum_{i \in [n]} a_i = 0$, and hence $N \in \mathbb{Z}_n$. 

69
This completes the proof of the claim. 

Since $M \otimes I_n \in \mathcal{L}_{\text{row}}$ and $I_n \otimes N \in \mathcal{L}_{\text{col}}$, we have from Claim 3.10 that $g_{\text{Det}_n} \subseteq \mathcal{L}_{\text{row}} + \mathcal{L}_{\text{col}}$. ■
Chapter 4

Equivalence testing for IMM

In this chapter we give a polynomial time randomized algorithm for equivalence testing of IMM. The contents of this chapter are from our work [KNST19].

4.1 Introduction

An \( m \) variate polynomial \( f \) is an affine projection of an \( n \) variate polynomial \( g \) via a full-rank transformation, if \( m \geq n \) and there is a full-rank matrix \( A \in \mathbb{F}^{n \times m} \) and a \( b \in \mathbb{F}^n \) such that \( f(x) = g(Ax + b) \) (see Definition 2.14 and the statements below the definition). In the affine projection via full-rank transformation problem, an \( m \) variate polynomial \( f \) and an \( n \) variate polynomial \( g \) is given as input (in some representation) where \( m \geq n \), and the task is to determine whether \( f \) is an affine projection of \( g \) via a full-rank transformation. Using variable reduction and translation equivalence test (described in Section 2.7) [Kay12a] reduced the affine projection via full-rank transformation problem in randomized polynomial time to equivalence testing that is when \( m = n \) and \( A \in \text{GL}(n, \mathbb{F}) \) (see also Section 4.1.1). The equivalence testing problem introduced in Section 1.1.4, is at least as hard as graph isomporhism, even when \( f \) and \( g \) are degree three polynomials given in the dense representation, that is as a list of coefficients. [Kay11, Kay12a, GGKS19] considered the affine projection projection via full-rank transformation problem when \( f \) is given as blackbox and \( g \) comes from a fixed polynomial family, and gave efficient randomized algorithms for it when \( g \) is an elementary symmetric polynomial/permanent/determinant/power symmetric polynomial or sum-of-products polynomial (see Section 1.1.4 for more details). In this chapter, we present the algorithms in Theorems 1.1a and 1.1b stated in Section 1.2.1 which checks whether a polynomial given as blackbox is an affine projection of \( \text{IMM}_{w,d} \) (see Section 2.3) via a full-rank transformation. We restate the theorems below and throughout this chapter assume \( \mathbb{F} = \mathbb{Q} \), but our results also hold over a finite field of large enough characteristic.
Theorem 1.1a: Given blackbox access to an $m$-variate polynomial $f \in \mathbb{F}[\mathbf{x}]$ of degree $d \in [5, m]$, the problem of checking if there exist a $\mathbf{w} \in \mathbb{N}^{d-1}$, a $\mathbf{B} \in \mathbb{F}^{n \times m}$ of rank $n$ equal to the number of variables in $\text{IMM}_w,d$, and a $\mathbf{b} \in \mathbb{F}^n$ such that $f = \text{IMM}_w,d(\mathbf{Bx} + \mathbf{b})$, can be solved in randomized $(m\beta)^{O(1)}$ time where $\beta$ is the bit length of the coefficients of $f$. Further, with probability at least $1 - n^{-\Omega(1)}$, the following is true: the algorithm returns a $\mathbf{w}$, a $\mathbf{B} \in \mathbb{F}^{n \times m}$ of rank $n$ such that $f = \text{IMM}_w,d(\mathbf{Bx} + \mathbf{b})$ if such $\mathbf{w}$, $\mathbf{B}$ and $\mathbf{b}$ exist, else it outputs ‘no such $\mathbf{w}$, $\mathbf{B}$ and $\mathbf{b}$ exist’.

Theorem 1.1b (Equivalence test for IMM) Given blackbox access to a homogeneous $n$-variate polynomial $f \in \mathbb{F}[\mathbf{x}]$ of degree $d \in [5, n]$, where $|\mathbf{x}| = n$, the problem of checking if there exist a $\mathbf{w} \in \mathbb{N}^{d-1}$ and an invertible $\mathbf{A} \in \mathbb{F}^{n \times n}$ such that $f = \text{IMM}_w,d(\mathbf{Ax})$, can be solved in randomized $(n\beta)^{O(1)}$ time where $\beta$ is the bit length of the coefficients of $f$. Further, with probability at least $1 - n^{-\Omega(1)}$ the following holds: the algorithm returns a $\mathbf{w}$, and an invertible $\mathbf{A} \in \mathbb{F}^{n \times n}$ such that $f = \text{IMM}_w,d(\mathbf{Ax})$ if such $\mathbf{w}$ and $\mathbf{A}$ exist, else it outputs ‘no such $\mathbf{w}$ and $\mathbf{A}$ exist’.

Detailed remarks on the above theorems are given in Section 1.2.1. Observe that if $f$ is an affine-projection of $\text{IMM}_w,d$ via a full-rank transformation then $f$ is computed by a full-rank ABP (see Definition 2.5) of width $w$ and length $d$, and hence the algorithm in Theorem 1.1a also reconstructs a full-rank ABP for $f$. Next, we present the algorithms in Theorems 1.1a and 1.1b, and the proof strategies for them.

4.1.1 Algorithm and proof strategy

The algorithm in Theorem 1.1a is given in Algorithm 3. At first, we trace the steps of this algorithm to show that proving Theorem 1.1a reduces to proving Theorem 1.1b using known methods. Then, we give an equivalence test for IMM in Algorithm 4, which is the contribution of the work presented in this chapter. Some relevant definitions, notations and concepts can be found in Chapter 2.

Reduction to equivalence test for IMM

We are given blackbox access to an $m$-variate polynomial $f(\mathbf{x})$ in Algorithm 3 where $\mathbf{x} = \{x_1, \ldots, x_m\}$. Suppose $f = \text{IMM}_w,d(B'\mathbf{x} + \mathbf{b}')$ for some unknown $w' \in \mathbb{N}^{d-1}$, $b' \in \mathbb{F}^n$ and $B' \in \mathbb{F}^{n \times m}$ of rank $n$, where $n$ is the number of variables in $\text{IMM}_{w',d}$.

(i) Variable reduction (Step 2): The number of essential/redundant variables of a polynomial remains unchanged under affine projection via full-rank transformation. Since

\[ A \text{ variable set } \mathbf{x} = \{x_1, \ldots, x_m\} \text{ is treated as a column vector } (x_1 \ldots x_m)^T \text{ in the expression } B\mathbf{x} + \mathbf{b}. \text{ The affine form entries of the column } B\mathbf{x} + \mathbf{b} \text{ are then plugged in place of the variables of } \text{IMM}_{w,d} \text{ (following a variable ordering, like the one mentioned in Section 2.3).} \]

72
\[ \text{IMM}_{w',d} \] has no redundant variables (follows easily from Claim 2.4), the number of essential variables of \( f \) equals \( n \). The algorithm eliminates the \( m - n \) redundant variables in \( f \) by using Algorithm 1 (in Section 2.7) and constructs a \( C \in \text{GL}(m) \) such that \( g = f(C\tilde{x}) \) has only the essential variables \( x = \{x_1, \ldots, x_n\} \). It follows that \( g = \text{IMM}_{w',d}(A'x + b') \), where \( A' \in \text{GL}(n) \) is the matrix \( B' \cdot C \) restricted to the first \( n \) columns.

(ii) **Equivalence test (Steps 5-9):** Since \( g = \text{IMM}_{w',d}(A'x + b') \), its \( d \)-th homogeneous component \( g^{[d]} = \text{IMM}_{w',d}(A'x) \). In other words, \( g^{[d]} \) is equivalent to \( \text{IMM}_{w',d} \) for an unknown \( w' \in \mathbb{N}^{d-1} \). At this point, the algorithm calls Algorithm 4 to find a \( w \) and an \( A \in \text{GL}(n) \) such that \( g^{[d]} = \text{IMM}_{w,d}(Ax) \), and this is achieved with high probability.

(iii) **Finding a translation (Steps 12-17):** Since \( g \) is equal to \( \text{IMM}_{w',d}(A' \cdot (x + A'^{-1}b')) = g^{[d]}(x + A'^{-1}b') \), \( g \) is translation equivalent to \( g^{[d]} \). With high probability, Algorithm 2 (in Section 2.7) finds an \( a \in \mathbb{F}^n \) such that \( g = g^{[d]}(x + a) \), implying \( g = \text{IMM}_{w,d}(Ax + Aa) \). Thus \( b = Aa \) is a valid translation vector.

(iv) **Final reconstruction (Steps 20-26):** From the previous steps, we have \( g = \text{IMM}_{w,d}(Ax + b) \). Although the variables \( \{x_{n+1}, \ldots, x_m\} \) are absent in \( g \), if we pretend that \( g \) is a polynomial in all the \( \tilde{x} \) variables then \( g = \text{IMM}_{w,d}(A_0\tilde{x} + b) \), where \( A_0 \) is an \( n \times m \) matrix such that the \( n \times n \) submatrix formed by restricting to the first \( n \) columns of \( A_0 \) equals \( A \) and the remaining \( m - n \) columns of \( A_0 \) have all zero entries. Hence \( f = g(C^{-1}\tilde{x}) = \text{IMM}_{w,d}(A_0C^{-1}\tilde{x} + b) \) which explains the setting \( B = A_0C^{-1} \) in step 20. The identity testing in steps 21-23 takes care of the situation when, to begin with, there are no \( w' \in \mathbb{N}^{d-1}, b' \in \mathbb{F}^n \) and \( B' \in \mathbb{F}^{m \times m} \) of rank \( n \) such that \( f = \text{IMM}_{w,d}(B'\tilde{x} + b') \).

**Equivalence test for IMM**

Algorithm 3 calls Algorithm 4 on a blackbox holding a homogeneous \( n \) variate polynomial \( f(x) \) of degree \( d \leq n \), and expects a \( w \in \mathbb{N}^{d-1} \) and an \( A \in \text{GL}(n) \) in return such that \( f = \text{IMM}_{w,d}(Ax) \), if such \( w \) and \( A \) exist. First, we argue that \( f \) can be assumed to be an irreducible polynomial.

(a) **Assuming irreducibility of input \( f \) in Algorithm 4:** The idea is to construct blackbox access to the irreducible factors of \( f \) using the efficient randomized polynomial factorization algorithm in Lemma 2.2, and and perform equivalence testing for IMM of smaller degree. This process succeeds with high probability and the details are as follows: If \( f \) is
Algorithm 3 Reconstructing a full-rank ABP

INPUT: Blackbox access to an \(m\)-variate polynomial \(f(\tilde{x})\) of degree \(d \leq m\).

OUTPUT: A full-rank ABP computing \(f\) if such an ABP exists.

1: /* Variable reduction */
2: Use Algorithm 1 to compute \(n\) and \(C \in \text{GL}(m)\) such that \(g = f(C\tilde{x})\) has only the essential variables \(x = \{x_1, \ldots, x_n\}\) of \(f\). If \(d > n\), output ‘\(f\) does not admit a full-rank ABP’ and stop.

3: /* Equivalence test: Finding \(w\) and \(A\) */
4: Construct a blackbox for \(g^{[d]}\), the \(d\)-th homogeneous component of \(g\) (see Section 2.7).
5: Use Algorithm 4 to find a \(w \in \mathbb{N}^{d-1}\) and an \(A \in \text{GL}(n)\) such that \(g^{[d]} = \text{IMM}_{w,d}(Ax)\).
6: if Algorithm 4 outputs ‘no such \(w\) and \(A\) exist’ then
7: Output ‘\(f\) does not admit a full-rank ABP’ and stop.
8: end if

9: /* Finding a translation \(b\) */
10: Use Algorithm 2 to find an \(a \in \mathbb{F}^n\) such that \(g = g^{[d]}(x + a)\).
11: if Algorithm 2 outputs ‘\(g\) is not translation equivalent to \(g^{[d]}\)’ then
12: Output ‘\(f\) does not admit a full-rank ABP’ and stop.
13: else
14: Set \(b = Aa\).
15: end if

16: /* Identity testing and final reconstruction */
17: Let \(A_0\) be the \(n \times m\) matrix obtained by attaching \(m - n\) ‘all-zero’ columns to the right of \(A\). Set \(B = A_0C^{-1}\).
18: Choose a point \(a \in S^m\) at random, where \(S \subseteq \mathbb{F}\) and \(|S| \geq n^{O(1)}\).
19: if \(f(a) \neq \text{IMM}_{w,d}(Ba + b)\) then
20: Output ‘\(f\) does not admit a full-rank ABP’ and stop.
21: else
22: Construct a full-rank ABP \(A\) of width \(w\) from \(B\) and \(b\). Output \(A\).
23: end if

Algorithm not square-free (which can be easily checked using Lemma 2.2) then \(f\) cannot be equivalent to \(\text{IMM}_{w,d}\) for any \(w\), as \(\text{IMM}_{w,d}\) is always square-free. Suppose \(f = f_1 \cdots f_k\), where \(f_1, \ldots, f_k\) are distinct irreducible factors of \(f\). If there are \(w' \in \mathbb{N}^{d-1}\) and \(A' \in \text{GL}(n)\) such that \(f = \text{IMM}_{w',d}(A'x)\), then the number of essential variables in \(f\) is \(n\) (as \(\text{IMM}_{w',d}\) has no redundant variables). Also, \(f_1 \cdots f_k = h_1(A'x) \cdots h_k(A'x)\) where \(h_1, \ldots, h_k\) are the irreducible factors of \(\text{IMM}_{w',d}\). The irreducible factors of \(\text{IMM}_{w',d}\) are ‘smaller IMMs’.
in disjoint sets of variables\footnote{Recall, $\text{IMM}_{w,d}$ is irreducible if $w_k > 1$ for every $k \in [d-1]$ where $w = (w_1, \ldots, w_{d-1})$.}. Let the degree of $f_\ell$ be $d_\ell$ and $n_\ell$ the number of essential variables in $f_\ell$. Then $n_1 + \ldots + n_k = n$. Now observe that if we invoke Algorithm 3 on input $f_\ell$, it calls Algorithm 4 from within on an irreducible polynomial, as $f_\ell$ is homogeneous and irreducible. Algorithm 3 returns a $w_\ell \in \mathbb{N}^{d_\ell - 1}$ and $B_\ell \in \mathbb{F}^{n_\ell \times n}$ of rank $n_\ell$ such that $f_\ell = \text{IMM}_{w_\ell, d_\ell}(B_\ell x)$ (ignoring the translation vector as $f_\ell$ is homogeneous). Let $w \in \mathbb{N}^{d-1}$ be the vector $(w_1 \ 1 \ w_2 \ 1 \ldots \ 1 \ w_k)^2$, and $A \in \mathbb{F}^{n \times n}$ such that the first $n_1$ rows of $A$ is $B_1$, next $n_2$ rows is $B_2$, and so on till last $n_k$ rows is $B_k$. Then, $f = \text{IMM}_{w,d}(Ax)$. Clearly, $A$ must be in $\text{GL}(n)$ as the number of essential variables of $f$ is $n$. Thus, it is sufficient to describe Algorithm 4 on an input $f$ that is irreducible.

(b) A comparison with [Kay12a] and our proof strategy: [Kay12a] gave equivalence tests for the permanent and determinant polynomials by making use of their Lie algebra (see Definition 2.17). Algorithm 4 also involves Lie algebra of IMM, but there are some crucial differences in the way Lie algebra is used in [Kay12a] and in here. The Lie algebra of permanent consists of diagonal matrices and hence commutative. By diagonalizing a basis of $g_f$ over $\mathbb{C}$, for an $f$ equivalent to permanent, we can reduce the problem to the much simpler permutation and scaling (PS) equivalence problem. The Lie algebra of $n \times n$ determinant, which is isomorphic to $\text{sl}_n \oplus \text{sl}_n$, is not commutative. However, a Cartan subalgebra of $\text{sl}_n$ consists of traceless diagonal matrices. This then helps reduce the problem to PS-equivalence by diagonalizing (over $\mathbb{C}$) a basis of the centralizer of a random element in $g_f$, for an $f$ equivalent to determinant. Both the equivalence tests involve simultaneous diagonalization of matrices over $\mathbb{C}$. It is a bit unclear how to carry through this step if the base field is $\mathbb{Q}$ and we insist on low bit complexity. The Lie algebra of IMM is not commutative. Also, we do not know if going to Cartan subalgebra helps, as we would like to avoid the simultaneous diagonalization step. Instead of Cartan subalgebras, we study invariant subspaces (Definition 2.10) of the Lie algebra $g_{\text{IMM}}$. A detailed analysis of the Lie algebra (in Section 3.1) reveals the structure of the irreducible invariant subspaces of $g_{\text{IMM}}$. It is observed that these invariant subspaces are intimately connected to the layer spaces (see Definition 2.6) of any full-rank ABP computing $f$. At a conceptual level, this connection helps us reconstruct a full-rank ABP. Once we have access to the layer spaces, we can retrieve the unknown width vector $w$ whence the problem reduces to the easier problem of reconstructing an almost set-multilinear ABP (Definition 4.1).
We now give some more details on Algorithm 4. Suppose there is a \( w \in \mathbb{N}^{d-1} \) such that \( f \) is equivalent to \( \text{IMM}_{w,d} \). The algorithm has four main steps:

(i) **Computing irreducible invariant subspaces (Steps 2-6):** The algorithm starts by computing a basis of the Lie algebra \( g_f \). It then invokes Algorithm 5 to compute bases of the \( d \) irreducible invariant subspaces of \( g_f \). Algorithm 5 works by picking a random element \( R' \) in \( g_f \) and factoring its characteristic polynomial \( h = g_1 \cdots g_s \). By computing the closure of vectors (Definition 2.11) picked from null spaces of \( g_1(R'), \ldots, g_s(R') \), the algorithm is able to find bases of the required invariant spaces.

(ii) **Computing layer spaces (Step 9):** The direct relation between the irreducible invariant spaces of \( g_{\text{hom}} \) and the layers spaces of any full-rank ABP computing \( f \) (as shown in Lemma 4.3) is exploited by Algorithm 7 to compute bases of these layer spaces. This also helps establish that all the layer spaces, except two of them, are ‘unique’ (see Lemma 4.2). The second and second-to-last layer spaces of a full-rank ABP are not unique; however the bigger space spanned by the first two layer spaces (similarly the last two layer spaces) is unique. Algorithm 7 finds bases for these two bigger spaces along with the \( d-2 \) remaining layer spaces.

(iii) **Reduction to almost set-multilinear ABP (Steps 12-15):** The layer spaces are then correctly reordered in Algorithm 8 using a randomized procedure to compute the appropriate evaluation dimensions (Definition 2.12). The reordering also yields a valid width vector \( w \). At this point, the problem easily reduces to reconstructing a full-rank almost set-multilinear ABP by mapping the bases of the layer spaces to distinct variables. This mapping gives an \( \hat{A} \in \text{GL}(n) \) such that \( f(\hat{A}x) \) is computable by a full-rank almost set-multilinear ABP of width \( w \). It is ‘almost set-multilinear’ (and not ‘set-multilinear’) as the second and the second-to-last layer spaces are unavailable; instead, two bigger spaces are available as mentioned above.

(iv) **Reconstructing a full-rank almost set-multilinear ABP (Steps 18-22):** Finally, we reconstruct a full-rank almost set-multilinear ABP computing \( f(\hat{A}x) \) using Algorithm 9. This algorithm is inspired by a similar algorithm for reconstructing set-multilinear ABP in [KS03], but it is a little different from the latter as we are dealing with an ‘almost’ set-multilinear ABP. The reconstructed ABP readily gives an \( A \in \text{GL}(n) \) such that \( f = \text{IMM}_{w,d}(Ax) \).

A final identity testing (Steps 25-30) takes care of the situation when, to begin with, there is no \( w \in \mathbb{N}^{d-1} \) that makes \( f \) equivalent to \( \text{IMM}_{w,d} \).
Algorithm 4 Equivalence test for IMM

INPUT: Blackbox access to a homogeneous \( n \) variate degree \( d \) polynomial \( f \) (which can be assumed to be irreducible without any loss of generality).
OUTPUT: A \( w \in \mathbb{N}^{d-1} \) and an \( A \in \text{GL}(n) \) such that \( f = \text{IMM}_{w,d}(Ax) \), if such \( w \) and \( A \) exist.
1: /* Finding irreducible invariant subspaces */
2: Compute a basis of the Lie algebra \( g_f \). (See Section 2.7.)
3: Use Algorithm 5 to compute the bases of the irreducible invariant subspaces of \( g_f \).
4: if Algorithm 5 outputs 'Fail' then
5: Output 'no such \( w \) and \( A \) exist' and stop.
6: end if
7: /* Finding layer spaces from irreducible invariant subspaces */
8: Use Algorithm 7 to compute bases of the layer spaces of a full-rank ABP computing \( f \), if such an ABP exists.
9: /* Reduction to almost set-multilinear ABP: Finding \( w \) */
10: Use Algorithm 8 to compute a \( w \in \mathbb{N}^{d-1} \) and an \( \hat{A} \in \text{GL}(n) \) such that \( h = f(\hat{A}x) \) is computable by a full-rank almost set-multilinear ABP of width \( w \).
11: if Algorithm 8 outputs 'Fail' then
12: Output 'no such \( w \) and \( A \) exist' and stop.
13: end if
14: /* Reconstructing an almost set-multilinear ABP: Finding \( A \) */
15: Use Algorithm 9 to reconstruct a full-rank almost set-multilinear ABP \( A' \) computing \( h \).
16: if Algorithm 9 outputs 'Fail' then
17: Output 'no such \( w \) and \( A \) exist' and stop.
18: end if
19: Replace the \( x \) variables in \( A' \) by \( \hat{A}^{-1}x \) to obtain a full-rank ABP \( A \). Compute \( A \in \text{GL}(n) \) from \( A \).
20: /* Final identity testing */
21: Choose a point \( a \in S^n \), where \( S \subseteq \mathbb{F} \) and \( |S| \geq n^{O(1)} \).
22: if \( f(a) \neq \text{IMM}_{w,d}(Aa) \) then
23: Output 'no such \( w \) and \( A \) exist' and stop.
24: else
25: Output \( w \) and \( A \).
26: end if

4.2 Almost set-multilinear ABP

In the proof of Theorem 1.1b, we eventually reduce the equivalence test problem to checking whether there exists an \( A \in \text{GL}(n) \), such that an input polynomial \( h(x) \) (given as blackbox)
equals $\text{IMM}_{w,d}(Ax)$, where $w$ is known, $x$ is the variables of $\text{IMM}_{w,d}$, and $A$ satisfies the following properties:

1. For all $k \in [d] \setminus \{2, d - 1\}$, the rows indexed by $x_k$ variables contain zero entries in columns indexed by variables other than $x_k$.

2. The rows indexed by $x_2$ and $x_{d-1}$ variables contain zero entries in columns indexed by variables other than $x_1 \cup x_2$ and $x_{d-1} \cup x_d$ respectively.

If there exists such a block-diagonal matrix $A$ then we say $h$ is computed by a **full-rank almost set-multilinear ABP** as defined below.

**Definition 4.1 (full-rank almost set-multilinear ABP)** A full-rank almost set-multilinear ABP of width $w = (w_1, w_2, \ldots, w_{d-1})$ and length $d$ is a product of $d$ matrices $X_1 \cdot X_2 \ldots X_d$, where $X_1$ is a row vector of size $w_1$, $X_d$ is a column vector of size $w_d - 1$ and $X_k$ is a $w_{k-1} \times w_k$ matrix for $k \in [2, d - 1]$. The entries in $X_k$ are linear forms in $x_k$ variables, for all $k \in [d] \setminus \{2, d - 1\}$, and the entries in $X_2$ and $X_{d-1}$ are linear forms in $x_1 \cup x_2$ and $x_{d-1} \cup x_d$ variables respectively, where $x_1 \cup x_2 \ldots \cup x_d$ is the set of variables in $\text{IMM}_{w,d}$.

Conventionally, in the definition of set-multilinear ABP, the entries of $X_i$ are linear forms in just $x_i$ variables – the ABP in the above definition is almost set-multilinear as matrices $X_2$ and $X_{d-1}$ violate this condition. An efficient randomized reconstruction algorithm for set-multilinear ABP follows from [KS03]. In order to apply a similar reconstruction algorithm to full-rank almost set-multilinear ABPs, we fix a canonical representation for the first two and the last two matrices as explained below.

**Canonical form or representation:** We say a full-rank almost set-multilinear ABP of width $w$ is in canonical form if the following hold:

1. $X_1 = (x_1^{(1)}, x_2^{(1)}, \ldots, x_{w_1}^{(1)})$,

2. the linear forms in $X_2$ are such that for $l, i \in [w_1]$ and $l < i$, the variable $x_i^{(1)}$ has a zero coefficient in the $(i, j)$-th entry (linear form) of $X_2$, where $j \in [w_2]$.

3. $X_d = (x_1^{(d)}, x_2^{(d)}, \ldots, x_{w_{d-1}}^{(d)})^T$,

4. the linear forms in $X_{d-1}$ are such that for $l, j \in [w_{d-1}]$ and $l < j$, the variable $x_l^{(d)}$ has a zero coefficient in the $(i, j)$-th entry (linear form) of $X_{d-1}$, where $i \in [w_{d-2}]$. 
The following claim states that for every full-rank almost set-multilinear ABP there is another ABP in canonical form computing the same polynomial, and the latter can be computed efficiently.

**Claim 4.1** Let $h$ be an $n$ variate, degree $d$ polynomial computable by a full-rank almost set-multilinear ABP of width $w = (w_1, w_2, \ldots, w_{d-1})$ and length $d$. There is a randomized algorithm that takes input blackbox access to $h$ and the width vector $w$, and outputs a full-rank almost set-multilinear ABP of width $w$ in canonical form computing $h$, with probability at least $1 - n^{-\Omega(1)}$. The running time of the algorithm is $(n\beta)^{O(1)}$, where $\beta$ is the bit length of the coefficients of $h$.

We prove the claim in Section 4.4.3. The algorithm is similar to reconstruction of set-multilinear ABP in [KS03], except that the latter needs to be adapted suitably as we are dealing with almost set-multilinear ABP.

### 4.3 Lie algebra of $f$ equivalent to IMM

Let $f$ be an $n$ variate polynomial such that $f = \text{IMM}_{w,d}(Ax)$, where $w = (w_1, w_2, \ldots, w_{d-1}) \in \mathbb{N}^{d-1}$ and $A \in \text{GL}(n)$. It follows, $n = w_1 + \sum_{i=2}^{d-1} w_{i-1}w_i + w_{d-1}$. We recall the notation defined in Section 3.1.3. The variables in IMM are identified with the unit vectors $e_1, e_2, \ldots, e_n$ in $\mathbb{F}^n$ – the vector $e_i$ corresponds to the $i$-th variable as per the variable ordering defined among variables. The unit vector $e_x$ corresponds to the variable $x$, and $\mathbb{U}_{1,2}$ denotes the coordinate subspace spanned by the unit vectors corresponding to the variables in $x_1 \oplus x_2$. Similarly $\mathbb{U}_k$ denotes the coordinate subspace spanned by the unit vectors corresponding to the variables in $x_k$ for $k \in [2, d-1]$, and $\mathbb{U}_{d-1,d}$ denotes the coordinate subspace spanned by the unit vectors corresponding to the variables in $x_{d-1} \oplus x_d$.

From Observation 2.2 and Lemma 3.6 we know $A^{-1}\mathbb{U}_{1,2}, A^{-1}\mathbb{U}_2, \ldots, A^{-1}\mathbb{U}_{d-1}, A^{-1}\mathbb{U}_{d-1,d}$ are the only irreducible invariant subspaces of $g_f$, and $A^{-1}\mathbb{U}_2$ (respectively $A^{-1}\mathbb{U}_{d-1}$) is the only invariant subspace properly contained in $A^{-1}\mathbb{U}_{1,2}$ (respectively $A^{-1}\mathbb{U}_{d-1,d}$). Also from Corollary 3.1 it follows that $\mathbb{F}^n = A^{-1}\mathbb{U}_{1,2} \oplus A^{-1}\mathbb{U}_3 \oplus \cdots \oplus A^{-1}\mathbb{U}_{d-2} \oplus A^{-1}\mathbb{U}_{d-1,d}$. In this section, we give an efficient randomized algorithm to compute a basis of each of the spaces $A^{-1}\mathbb{U}_{1,2}, A^{-1}\mathbb{U}_2, \ldots, A^{-1}\mathbb{U}_{d-1}, A^{-1}\mathbb{U}_{d-1,d}$ given only blackbox access to $f$ (but no knowledge of $w$ or $A$).

#### 4.3.1 Computing invariant subspaces of the Lie algebra $g_f$

First, we efficiently compute a basis $\{L'_1, L'_2, \ldots, L'_m\}$ of $g_f$ using the algorithm stated in Lemma 2.4. By Claim 2.1, $L_1 = AL'_1A^{-1}, L_2 = AL'_2A^{-1}, \ldots, L_m = AL'_mA^{-1}$ form a basis of
Figure 4.1: Random element $R$ in $g_{\text{MM}}$

$g_{\text{MM}}$. Suppose $R' = \sum_{i=1}^{m} r_i L_i'$ is a random element of $g_f$, chosen by picking the $r_i$’s independently and uniformly at random from $[2n^3]$. Then $R = AR' A^{-1} = \sum_{i=1}^{m} r_i L_i$ is a random element of $g_{\text{MM}}$ and it follows from Lemma 3.5 that the characteristic polynomial of $R$ is square-free with probability at least $1 - n^{-\Omega(1)}$. So assume henceforth that the characteristic polynomial of $R$ (and hence also of $R'$) is square-free.

Moreover, from Figure 3.1 it follows that $R$ has the structure as shown in Figure 4.1. Let $h(x) = \prod_{i=1}^{d} h_i(x)$ be the characteristic polynomial of $R$, and $g_1(x), g_2(x), \ldots, g_s(x)$ be the distinct irreducible factors of $h(x)\!$ over $\mathbb{F}$. Suppose $N_i$ is the null space of $g_i(R)\!$. Thus $N_i$, the null space of $g_i(R)$ (equal to $A \cdot g_i(R') \cdot A^{-1}$), is $AN_i$ for $i \in [s]$. We study the null spaces $N_1, N_2, \ldots, N_s$ in the next two claims and show how to extract out the irreducible invariant subspaces of $g_f$ from $N_1, N_2, \ldots, N_s$ (as specified in Algorithm 5).

**Claim 4.2** For all $i \in [s]$, let $N_i$ and $N_i'$ be the null spaces of $g_i(R)$ and $g_i(R')$. Then

1. $\mathbb{F}^n = N_1 \oplus N_2 \oplus \cdots \oplus N_s = N'_1 \oplus N'_2 \oplus \cdots \oplus N'_s$.

2. For all $i \in [s]$, $\dim(N_i) = \dim(N_i') = \deg_x(g_i)$.

**Proof:** Since $N_i' = A^{-1}N_i$ and $A^{-1} \in \text{GL}(n)$, it is sufficient to show $\mathbb{F}^n = N_1 \oplus N_2 \oplus \cdots \oplus N_s$ and $\dim(N_i) = \deg_x(g_i)$. Further, observe that each subspace $N_i$ is non-trivial – if $N_i = \{0\}$ then for all $v \in \mathbb{F}^n$, $h(R) \cdot v = g_1(R)g_2(R) \cdots g_s(R) \cdot v = 0$ implying $g_2(R) \cdots g_s(R) \cdot v = 0$. As the characteristic polynomial and the minimal polynomial have the same irreducible factors
this gives a contradiction.

To show the sum of \( N_i \)'s is a direct sum it is sufficient to show the following: if \( \sum_{l=1}^{s} u_l = 0 \) where \( u_l \in N_l \) then \( u_l = 0 \) for \( l \in [s] \). Define for \( i \in [s] \)

\[
\hat{g}_i := \prod_{j=1, j \neq i}^{s} g_j(x) = \frac{h(x)}{g_i(x)}. \tag{4.1}
\]

Since \( \hat{g}_i(R) \cdot u_j = 0 \) for \( j \neq i \),

\[
\hat{g}_i(R) \cdot \left( \sum_{l=1}^{s} u_l \right) = \hat{g}_i(R) \cdot u_i = 0. \tag{4.2}
\]

As \( g_i(x) \) and \( \hat{g}_i(x) \) are coprime polynomials, there are \( p_i(x), q_i(x) \in \mathbb{F}[x] \) such that

\[
p_i(x)g_i(x) + q_i(x)\hat{g}_i(x) = 1 \Rightarrow p_i(R)g_i(R) + q_i(R)\hat{g}_i(R) = I_n \Rightarrow (p_i(R)g_i(R)) \cdot u_i + (q_i(R)\hat{g}_i(R)) \cdot u_i = u_i.
\]

Both \( (p_i(R)g_i(R)) \cdot u_i = 0 \) (as \( u_i \in N_i \)) and \( (q_i(R)\hat{g}_i(R)) \cdot u_i = 0 \) (by Equation (4.2)). Hence \( u_i = 0 \) for all \( i \in [s] \).

Let \( \tilde{R} \) be the linear the linear map \( R \) restricted to the subspace \( N_i \) (this is well defined as \( N_i \) is an invariant subspace of \( R \)). Then, \( g_i(\tilde{R}) = 0 \). Since \( g_i \) is irreducible, from Cayley-Hamilton theorem it follows that \( g_i \) divides the characteristic polynomial of \( \tilde{R} \) implying \( \deg_x(g_i) \leq \dim(N_i) \). As a consequence, we have

\[
n = \sum_{i=1}^{s} \deg_x g_i \leq \sum_{i=1}^{s} \dim N_i \leq \dim \mathbb{F}^n = n. \tag{4.3}
\]

Each inequality is an equality, which proves the claim.

\[\blacksquare\]

**Claim 4.3** Suppose \( g_i(x) \) is an irreducible factor of the characteristic polynomial \( h_k(x) \) of \( R_k \) (depicted in Figure 4.1) for some \( k \in [d] \). Then the following holds:

1. If \( k \in [2, d - 1] \) then \( N_i \subseteq U_k \) (equivalently \( N_i' \subseteq A^{-1}U_k \)).

2. If \( k = 1 \) then \( N_i \subseteq U_{1,2} \) (equivalently \( N_i' \subseteq A^{-1}U_{1,2} \)), and if \( k = d \) then \( N_i \subseteq U_{d-1,d} \) (equivalently \( N_i' \subseteq A^{-1}U_{d-1,d} \)).
Proof: Figure 4.2 depicts the matrix $h_k(R)$ and as shown in it, call the submatrix restricted to the rows labelled by variables in $x_2$ and columns labelled by variables in $x_1 \cup x_2$, $M_{k,2}$; define $M_{k,d-1}$ similarly. Let $v \in N_i$. For every $j \in [d]$, let $v_j$ be the subvector of $v$ restricted to the rows labelled by variables in $x_j$, and $v_{1,2}$ (respectively $v_{d-1,d}$) be the subvector of $v$ restricted to the rows labelled by variables in $x_1 \cup x_2$ (respectively $x_{d-1} \cup x_d$). Since $v \in N_i$, $g_i(R) \cdot v = 0$ implying $h_k(R) \cdot v = 0$. Thus we have the following set of equations:

$$
\begin{align*}
  h_k(R_1) \cdot v_1 &= 0 \\
  M_{k,2} \cdot v_{1,2} &= 0 \\
  h_k(R_j) \cdot v_j &= 0 \quad \text{for } j \in [3, d - 2] \\
  M_{k,d-1} \cdot v_{d-1,d} &= 0 \\
  h_k(R_d) \cdot v_d &= 0.
\end{align*}
$$

(4.4)

Case a: $k \in [2, d - 1]$; since $h_j(x)$ is the characteristic polynomial of $R_j$, $h_j(R_j) = 0$ implying $h_j(R_j) \cdot v_j = 0$ for every $j \in [d]$. As $k \neq 1$, $h_k(x)$ and $h_1(x)$ are coprime and from Equation (4.4) $h_k(R_1) \cdot v_1 = 0$. Hence, $v_1 = 0$ and for a similar reason $v_d = 0$ as $k \neq d$. Thus in Equation (4.4) we have

$$
\begin{align*}
  M_{k,2} \cdot v_{1,2} &= h_k(R_2) \cdot v_2 \quad \text{= 0} \\
  M_{k,d-1} \cdot v_{d-1,d} &= h_k(R_{d-1}) \cdot v_{d-1} \quad \text{= 0}.
\end{align*}
$$

Therefore for every $j \in [d]$, $h_k(R_j) \cdot v_j = 0$. If $j \neq k$ then $h_j(x)$ and $h_k(x)$ are coprime, thus from $h_j(R_j) \cdot v_j = 0$ we infer $v_j = 0$ and hence $v \in U_k$. 

Figure 4.2: Matrix $h_k(R)$
Proof: If \( h_k(R_d) \cdot v_d = 0, h_d(R_d) \cdot v_d = 0, \) and \( h_k(x), h_d(x) \) are coprime, we get \( v_d = 0. \) Hence from Equation (4.4),

\[
M_{k,d-1} \cdot v_{d-1,d} = h_k(R_{d-1}) \cdot v_{d-1} = 0.
\]

Again for \( j \in [3, d], h_k(R_j) \cdot v_j = 0 \) and \( h_j(x), h_k(x) \) are coprime for every \( j \neq k. \) Hence \( v_j = 0 \) for \( j \in [3, d] \) implying \( v \in U_{1,2}. \)

Claim 4.4 1. If \( g_l(x), g_2(x), \ldots, g_r(x) \) are all the irreducible factors of \( h_k(x) \) for \( k \in [2, d-1] \) then \( A^{-1}U_k = N'_{l_1} \oplus N'_{l_2} \oplus \cdots \oplus N'_{l_r}. \)

2. If \( g_l(x), g_2(x), \ldots, g_r(x) \) are all the irreducible factors of \( h_1(x)h_2(x) \) (respectively \( h_{d-1}(x)h_d(x) \)) then \( A^{-1}U_{1,2} = N'_{l_1} \oplus N'_{l_2} \oplus \cdots \oplus N'_{l_r} \) (respectively \( A^{-1}U_{d-1,d} = N'_{l_1} \oplus N'_{l_2} \oplus \cdots \oplus N'_{l_r}. \))

Proof: If \( k \in [2, d-1] \) then \( N'_{l_1} \oplus N'_{l_2} \oplus \cdots \oplus N'_{l_r} \) is a direct sum and

\[
dim(A^{-1}U_k) = \deg_x(h_k) = \sum_{j=1}^r \deg_x(g_j) = \sum_{j=1}^r \dim(N'_{l_j}), \text{ which follow from Claim 4.2.}
\]

Hence from Claim 4.3, \( A^{-1}U_k = N'_{l_1} \oplus N'_{l_2} \oplus \cdots \oplus N'_{l_r}. \) The proof for the second part is similar.

Lemma 4.1 Given as input bases of the null spaces \( N'_1, N'_2, \ldots, N'_r \) we can compute bases of the spaces \( A^{-1}U_{1,2}, A^{-1}U_{2,1}, \ldots, A^{-1}U_{d-1}, A^{-1}U_{d-1,d} \) in deterministic polynomial time.

Proof: Recall \( N'_{l_i} \) is the null space of \( g_i(R^l) \), where \( g_i(x) \) is an irreducible factor of \( h_k(x) \) for some \( k \in [d]. \)

Case A: \( k \in [2, d-1] \); from Claim 4.3 it follows that \( N'_{l_i} \subseteq A^{-1}U_k. \) Pick a basis vector \( v \) in \( N'_{l_i} \) and compute the closure of \( v \) under the action of \( g_f \) using Algorithm 6 given in Section 4.3.2. Since the closure of \( v \) is the smallest invariant subspace of \( g_f \) containing \( v \), by Claim 3.6 the closure of \( v \) equals \( A^{-1}U_k. \)

Case B: \( k = 1 \) or \( k = d; \) the arguments for \( k = 1 \) and \( k = d \) are similar. We prove it for \( k = 1. \) From Claim 4.3 we have \( N'_{l_i} \subseteq A^{-1}U_{1,2}. \) Pick a basis vector \( v \) of \( N'_{l_i} \) and compute its closure under the action of \( g_f \) using Algorithm 6. Similar to case A, this gives us an invariant subspace of \( g_f \) contained in \( A^{-1}U_{1,2} \) and by Claim 3.7 this invariant subspace is either \( A^{-1}U_2 \) or \( A^{-1}U_{1,2}. \) However, \( N'_{l_i} \cap A^{-1}U_2 \) (by Claim 4.4) is empty, as \( g_i(x) \) is an irreducible factor of
4.1, we find the invariant subspaces of these null spaces. We present this formally in Algorithm 5.

**Comments on Algorithm 5:**

a. Observe that in step 6 of the algorithm we need \( \mathbb{F} \) to be \( \mathbb{Q} \) (as assumed) or a finite field because univariate factorization can be done effectively over such fields [LLL82, Ber67, CZ81].

b. When Algorithm 5 is invoked in Algorithm 4 for an \( n \) variate degree \( d \) polynomial \( f \), there may not exists a \( w \in \mathbb{N}^{d-1} \) and an \( A \in \text{GL}(n) \) such that \( f = \text{IMM}_{w,d}(Ax) \). We point out a few additional checks that need to be added to the above algorithm to handle this case. In step 9, if the pruned list (after removing repetitions) has size other than \( d \) then output ‘Fail’. Also from Claim 3.7, exactly two subspaces in the pruned list \( \{V_1, V_2, \ldots, V_d\} \), say \( V_2 \) and \( V_{d-1} \), should be subspaces of other vector spaces, say \( V_1 \)

---

**Algorithm 5** Computing irreducible invariant subspaces of \( g_f \)

**INPUT:** A basis \( \{L'_1, L'_2, \ldots, L'_n\} \) of \( g_f \).

**OUTPUT:** Bases of the irreducible invariant subspaces of \( g_f \).

1. Pick a random element \( R' = \sum_{j=1}^m r_j L'_j \) in \( g_f \), where \( r_j \in R \ [2n^3] \).
2. Compute the characteristic polynomial \( h(x) \) of \( R' \).
3. **if** \( h(x) \) is not square-free **then**
   4. Output 'Fail' and stop.
5. **end if**
6. Factor \( h(x) = g_1(x) \cdot g_2(x) \ldots g_s(x) \) into irreducible factors over \( \mathbb{F} \).
7. Find bases of the null spaces \( N'_1, N'_2, \ldots, N'_s \) of \( g_1(R'), g_2(R'), \ldots, g_s(R') \) respectively.
8. For every \( N'_i \), pick a vector \( v \) in the basis of \( N'_i \) and compute the closure of \( v \) with respect to \( g_f \) using Algorithm 6 given in Section 4.3.2.
9. Let \( \{V_1, V_2, \ldots, V_s\} \) be the list of the closure spaces; check for all \( i \neq j \) and \( i, j \in [s] \), whether \( V_i = V_j \) to remove repetitions from the above list and get the pruned list \( \{V_1, V_2, \ldots, V_d\}^1 \).
10. Output the set \( \{V_1, V_2, \ldots, V_d\} \).

---

To summarize, first we pick a random element \( R' \) in \( g_f \), find its characteristic polynomial \( h(x) \) and factorize \( h(x) \) to get the irreducible factors \( g_1(x), g_2(x), \ldots, g_s(x) \). Then we compute the null spaces \( N'_1, N'_2, \ldots, N'_s \) of \( g_1(R'), g_2(R'), \ldots, g_s(R') \) respectively. By applying Lemma 4.1, we find the invariant subspaces of \( g_f \), \( A^{-1}u_{1,2}, A^{-1}u_2, \ldots, A^{-1}u_{d-1}, A^{-1}u_{d-1,d} \) from these null spaces. We present this formally in Algorithm 5.
and $\mathcal{V}_d$ respectively. We can find these two spaces by doing a pairwise check among the $d$ vector spaces. If such subspaces do not exist among $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_d$ then output 'Fail'. Further, if $\mathbb{F}^n \neq \mathcal{V}_1 \oplus \mathcal{V}_3 \oplus \cdots \oplus \mathcal{V}_{d-2} \oplus \mathcal{V}_d$ (assuming $\mathcal{V}_2 \subseteq \mathcal{V}_1$ and $\mathcal{V}_{d-1} \subseteq \mathcal{V}_d$) then output 'Fail'.

c. It follows from the above discussion, if $f = \text{IMM}_{w,d}(Ax)$ then we can assume $\mathcal{V}_3, \mathcal{V}_4, \ldots, \mathcal{V}_{d-2}$ are the spaces $A^{-1}\mathcal{U}_3, A^{-1}\mathcal{U}_4, \ldots, A^{-1}\mathcal{U}_{d-2}$ in some unknown order. The spaces $\mathcal{V}_1, \mathcal{V}_2$ and $\mathcal{V}_d, \mathcal{V}_{d-1}$ are either the spaces $A^{-1}\mathcal{U}_{1,2}, A^{-1}\mathcal{U}_2$ and $A^{-1}\mathcal{U}_{d-1,2}, A^{-1}\mathcal{U}_{d-1}$ respectively, or the spaces $A^{-1}\mathcal{U}_{d-1,1}, A^{-1}\mathcal{U}_{d-1}$ and $A^{-1}\mathcal{U}_{1,2}, A^{-1}\mathcal{U}_2$ respectively.

4.3.2 Closure of a vector under the action of $g_f$

Algorithm 6 computes the closure of $v \in \mathbb{F}^n$ under the action of a space $\mathcal{L}$ spanned by $n \times n$ matrices. Let $\{M_1, M_2, \ldots, M_m\}$ be a basis of $\mathcal{L}$ where $M_i \in \mathbb{F}^{n \times n}$. For a set of vectors $T = \{v_1, v_2, \ldots, v_q\} \subseteq \mathbb{F}^n$, let $\mathcal{L} \cdot T$ denote the set $\{M_a v_b \mid a \in [m] \text{ and } b \in [q]\}$.

**Algorithm 6** Computing the closure of $v$ under the action of $\mathcal{L}$

| INPUT: $v \in \mathbb{F}^n$ and a basis $\{M_1, M_2, \ldots, M_m\}$ of $\mathcal{L}$. |
| OUTPUT: Basis of the closure of $v$ under the action of $\mathcal{L}$. |
| 1: Let $V^{(0)} = \{v\}$ and $V^{(1)} = \text{span}_F\{v, M_1 v, \ldots, M_m v\}$. |
| 2: Set $i = 1$. |
| 3: Compute a basis of $V^{(1)}$ and let $T_1 = \{v_1, v_2, \ldots, v_{q_1}\}$ be this basis. |
| 4: while $V^{(i-1)} \neq V^{(i)}$ do |
| 5: Set $i = i + 1$. |
| 6: Compute a basis for $V^{(i)} = \text{span}_F\{T_{i-1} \cup \mathcal{L} \cdot T_{i-1}\}$ and let $T_i = \{v_1, v_2, \ldots, v_{q_i}\}$ be this basis. |
| 7: end while |
| 8: Output $T_i$. |

**Claim 4.5** Algorithm 6 computes the closure of $v \in \mathbb{F}^n$ under the action of $\mathcal{L}$ in time polynomial in $n$ and the bit length of the entries of $v$ and $M_1, M_2, \ldots, M_m$.

**Proof:** The closure of $v$ under the action of $\mathcal{L}$ is the $\mathbb{F}$-linear span of all vectors of the form $\mu.v$, where $\mu$ is a non-commutative monomial in $M_1, M_2, \ldots, M_m$ (including unity). Algorithm 6 computes exactly this set and hence the closure of $v$. Moreover, $\dim(V^{(i)}) \leq n$ and in every iteration of the while loop $\dim(V^{(i)}) > \dim(V^{(i-1)})$, until $V^{(i)} = V^{(i-1)}$. Hence, Algorithm 6 runs in time polynomial in $n$ and the bit length of the entries of $v$ and $M_1, M_2, \ldots, M_m$. 

85
4.4 Reconstruction of full-rank ABP for $f$

Let $f$ be a polynomial equivalent to $\text{IMM}_{w,d}$ for some (unknown) $w \in \mathbb{N}^{d-1}$. In this section, we show that the invariant subspaces of $g_f$ let us compute a $w \in \mathbb{N}^{d-1}$ and an $A \in \text{GL}(n)$ such that $f = \text{IMM}_{w,d}(Ax)$. Since $f$ is equivalent to $\text{IMM}_{w,d}$, it is computable by a full-rank ABP $X_1 \cdot X_2 \cdots X_{d-1} \cdot X_d$ of width $w$ and length $d$ with linear form entries in the matrices. We call this full-rank ABP $\mathcal{A}$ which, as explained below, is not the only full-rank ABP computing $f$.

Many full-rank ABPs for $f$: The full-rank ABP $X_1' \cdot X_2' \cdots X_d'$ resulting from each of the following three transformations on $\mathcal{A}$ still computes $f$,

1. **Transposition:** Set $X_k' = X_{d+1-k}^T$ for $k \in [d]$.
2. **Left-right multiplications:** Let $A_1, \ldots, A_{d-1}$ be matrices such that $A_k \in \text{GL}(w_k)$ for every $k \in [d-1]$. Set $X_1' = X_1 \cdot A_1$, $X_d' = A_{d-1}^{-1} \cdot X_d$, and $X_k' = A_{k-1}^{-1} \cdot X_k \cdot A_k$ for $k \in [2,d-1]$.
3. **Corner translations:** Suppose \{C_{11}, C_{12}, \ldots, C_{w_2}\} and \{C_{d1}, C_{d2}, \ldots, C_{dw_{d-2}}\} are two sets containing anti-symmetric matrices in $\mathbb{F}^{w_1 \times w_2}$ and $\mathbb{F}^{w_{d-1} \times w_{d-1}}$ respectively. Let $Y_2 \in \mathbb{F}[x]^{w_1 \times w_2}$ (respectively $Y_{d-1} \in \mathbb{F}[x]^{w_{d-2} \times w_{d-1}}$) be a matrix with its $i$-th column (respectively $i$-th row) equal to $C_{1i} \cdot X_1^T$ (respectively $X_d^T \cdot C_{di}$). Set $X_2' = X_2 + Y_2$, $X_{d-1}' = X_{d-1} + Y_{d-1}$, and $X_k' = X_k$ for $k \in [d] \setminus \{2,d-1\}$.

In each of the above three cases $f = X_1' \cdot X_2' \cdots X_d'$; this is easy to verify for cases 1 and 2, in case 3 observe that $X_1 \cdot C_{1i} \cdot X_1^T = X_d^T \cdot C_{di} \cdot X_d = 0$.

It turns out that the full-rank ABPs obtained by (repeatedly) applying the above three transformations on $\mathcal{A}$ are the only full-rank ABPs computing $f$. This would follow from the discussion in Chapter 5. Although there are multiple full-rank ABPs for $f$, the layer spaces of these ABPs are unique (Lemma 4.2). This uniqueness of the layer spaces essentially facilitates the recovery of a full-rank ABP for $f$. Let us denote the span of the linear forms$^1$ in $X_1$ and $X_2$ (respectively $X_{d-1}$ and $X_d$) by $\mathcal{X}_{1,2}$ (respectively $\mathcal{X}_{d-1,d}$).

**Lemma 4.2 (Uniqueness of the layer spaces of full-rank ABP for $f$)** Suppose $X_1 \cdot X_2 \cdots X_d$ and $X_1' \cdot X_2' \cdots X_d'$ are two full-rank ABPs of widths $w = (w_1, w_2, \ldots, w_{d-1})$ and $w' = (w_1', w_2', \ldots, w'_{d-1})$ respectively, computing the same polynomial $f$. Then one of the following two cases is true:

a. $w_k' = w_k$ for $k \in [d-1]$, and the spaces $\mathcal{X}_1, \mathcal{X}_{1,2}, \mathcal{X}_3, \ldots, \mathcal{X}_{d-1,d}, \mathcal{X}_d'$ are the spaces $\mathcal{X}_{1,1,2,3, \ldots, d-1,d}, \mathcal{X}_d$ respectively.

---

$^1$Identify linear forms with vectors in $\mathbb{F}^n$ as mentioned in Definition 2.6.
b. \( w'_k = w_{d-k} \) for \( k \in [d - 1] \), and the spaces \( X'_1, X'_1, X'_2, \ldots, X'_{d-1}, X'_d \) are the spaces \( X_d, X_{d-1}, X_{d-2}, \ldots, X'_2, X'_1 \) respectively.

The lemma would help characterize the group of symmetries of IMM in Chapter 5; the proof would follow readily from Claim 4.6 in Section 4.4.2. With an eye on Chapter 5 and for better clarity in the reduction to almost set-multilinear ABP in Section 4.4.2, we take a slight detour and show next how to compute these 'unique' layer spaces of \( A \).

### 4.4.1 Computing layer spaces from invariant subspaces of \( g_f \)

Algorithm 5 outputs bases of the irreducible invariant subspaces \( \{ V_i | i \in [d] \} \) of \( g_f \). Recall, we assumed without loss of generality that \( V_2 \) and \( V_{d-1} \) are subspaces of \( V_1 \) and \( V_d \) respectively. The spaces \( V_1, V_2 \) and \( V_d, V_{d-1} \) are either the spaces \( A^{-1}U_{1,2}, A^{-1}U_2 \) and \( A^{-1}U_{d-1,d}, A^{-1}U_{d-1} \) respectively, or the spaces \( A^{-1}U_{d-1,d}, A^{-1}U_{d-1} \) and \( A^{-1}U_1, A^{-1}U_d \) respectively. Every other \( V_k \) is equal to \( A^{-1}U_{\sigma(k)} \) for some permutation \( \sigma \) on \([3, d - 2]\) (\( \sigma \) is not known at the end of Algorithm 5). Hence,

\[
\mathbb{F}^n = V_1 \oplus V_3 \oplus \cdots \oplus V_{d-2} \oplus V_d. \tag{4.5}
\]

Since \( V_2 \subseteq V_1 \), we can start with a basis of \( V_2 \) and fill in more elements from the basis of \( V_1 \) to get a new basis of \( V_1 \). Thus we can assume the basis of \( V_2 \) is contained in the basis of \( V_1 \). Likewise, the basis of \( V_{d-1} \) is contained in the basis of \( V_d \).

Order the basis vectors of \( V_1 \) such that the basis vectors of \( V_2 \) are at the end and order the basis vectors of \( V_d \) such that the basis vectors of \( V_{d-1} \) are at the beginning. For \( k \in [3, d - 2] \), the basis vectors of \( V_k \) are ordered in an arbitrary way. Let \( u_k \) denote the dimension of \( V_k \) for \( k \in [d] \). We identify the space \( V_k \) with an \( n \times u_k \) matrix \( V_k \), where the \( i \)-th column in \( V_k \) is the \( i \)-th basis vector of \( V_k \) in the above specified order. Algorithm 7 computes the layer spaces of \( A \) using \( V_1 \) to \( V_d \). Let \( t_2 = u_1 \) and \( t_k = u_k + t_{k-1} \) for \( k \in [3, d - 2] \).

**Comments on Algorithm 7:** Algorithm 4 invokes Algorithm 7 only after Algorithm 5, which returns 'Fail' if \( \mathbb{F}^n \neq V_1 \oplus V_3 \oplus \cdots \oplus V_{d-2} \oplus V_d \) (see comments after Algorithm 5). This ensures Equation (4.5) is satisfied and so \( V^{-1} \) exists in step 2 of the above algorithm, even if there are no \( w \in \mathbb{N}^{d-1} \) and \( A \in \text{GL}(n) \) such that \( f = \text{IMM}_{w,d}(Ax) \).

**Lemma 4.3** If \( f = X_1, X_2 \cdots X_d \) and \( y_1, y_1, y_2, y_3, \ldots, y_{d-2}, y_{d-1}, y_d \) is the output of Algorithm 7 then there is a permutation \( \sigma \) on \([3, d - 2]\) such that the following hold:

1. For every \( k \in [3, d - 2] \), \( y_k = X_{\sigma(k)} \).

87
**Algorithm 7** Computing the layer spaces of $A$

**INPUT:** Bases of the irreducible invariant subspaces of $g_f$.
**OUTPUT:** Bases of the layer spaces of $A$.

1. Form an $n \times n$ matrix $V$ by concatenating the columns of the matrices $V_1, V_3, \ldots, V_{d-2}, V_d$ in order, that is $V = [V_1 | V_3 | \ldots | V_{d-2} | V_d]$.
2. Compute $V^{-1}$. Number the rows of $V^{-1}$ by 1 to $n$.
3. Let $y_1$ be the space spanned by the first $u_1 - u_2$ rows of $V^{-1}$, and $y_{1,2}$ be the space spanned by the first $u_1$ rows of $V^{-1}$. Let $y_{d-1,d}$ be the space spanned by the last $u_d$ rows of $V^{-1}$ and $y_d$ be the space spanned by the last $u_d - u_{d-1}$ rows of $V^{-1}$. Finally, for every $k \in [3, d - 2]$, let $y_k$ be the space spanned by the rows of $V^{-1}$ that are numbered by $t_{k-1} + 1$ to $t_{k-1} + u_k$.

Output the bases of the spaces $y_1, y_{1,2}, y_3, \ldots, y_{d-2}, y_{d-1,d}, y_d$ in order.

2. Either $y_1, y_{1,2}$ and $y_d, y_{d-1,d}$ are $x_1, x_{1,2}$ and $x_d, x_{d-1,d}$ respectively, or $y_1, y_{1,2}$ and $y_d, y_{d-1,d}$ are $x_d, x_{d-1,d}$ and $x_1, x_{1,2}$ respectively.

**Proof:** Assume $V_1$ and $V_d$ are the spaces $A^{-1}U_{1,2}$ and $A^{-1}U_{d-1,d}$ respectively. In this case we will show $y_1, y_{1,2}$ and $y_d, y_{d-1,d}$ are $x_1, x_{1,2}$ and $x_d, x_{d-1,d}$ respectively\(^{1}\). Hence, $u_1 = w_1 + w_1 w_2, u_2 = w_1 w_2, u_{d-1} = w_1 w_2 w_{d-1}$ and $u_d = w_1 w_2 w_{d-1}$. From the order of the columns in $V_1$ and $V_d$ we have $V_1 = A^{-1}E_1$ and $V_d = A^{-1}E_d$, where $E_1$ and $E_d$ are $n \times u_1$ and $n \times u_d$ matrices respectively and they look as shown in Figure 4.3. The rows of $E_1$ and $E_d$ are labelled by $n$ variables in $x_1$ to $x_d$, whereas the columns of $E_1$ are labelled by variables in $x_1$ and $x_2$ and the columns of $E_d$ are labelled by variables in $x_{d-1}$ and $x_d$. Moreover, the nonzero

\(^{1}\)If $V_1$ and $V_d$ are the spaces $A^{-1}U_{d-1,d}$ and $A^{-1}U_{1,2}$ respectively, then $y_1, y_{1,2}$ and $y_d, y_{d-1,d}$ are $x_d, x_{d-1,d}$ and $x_1, x_{1,2}$ respectively – the proof of this case is similar.

---

Figure 4.3: Matrices $E_1$, $E_d$ and $E_k$
entries in these matrices are restricted to the shaded region in Figure 4.3.

For \( k \in [3, d-2] \), \( V_k = A^{-1}U_{\sigma(k)} \) where \( \sigma \) is a permutation on \( [3, d-2] \). Hence, \( u_k = w_{\sigma(k)-1}w_{\sigma(k)} \) and \( V_k = A^{-1}E_k \) where \( E_k \) is a \( n \times u_k \) matrix and looks as shown in Figure 4.3. Again the rows of \( E_k \) are labelled by the variables \( x_1 \) to \( x_d \), whereas the columns of \( E_k \) are labelled by variables in \( x_{\sigma(k)} \). The nonzero entries in \( E_k \) are restricted to the shaded region in Figure 4.3 whose rows are labelled by variables in \( x_{\sigma(k)} \). Let \( E \) be the concatenation of these matrices, \( E = [E_1 \mid E_3 \mid E_4 \mid \ldots \mid E_{d-2} \mid E_d] \). The rows of \( E \) are labelled by \( x_1, x_2, \ldots, x_d \) as usual, but now the columns are labelled by \( x_1, x_2, x_{\sigma(3)}, \ldots, x_{\sigma(d-2)}, x_{d-1}, x_d \) in order as shown in Figure 4.4. Then \( V = A^{-1}E \) implying \( V^{-1} = E^{-1}A \). Owing to the structure of \( E \), \( E^{-1} \) looks as shown in Figure 4.4. The rows of \( E^{-1} \) are labelled by \( x_1, x_2, x_{\sigma(3)}, \ldots, x_{\sigma(d-2)}, x_{d-1}, x_d \) in order, whereas the columns are labelled by the usual ordering \( x_1, x_2, \ldots, x_d \). The submatrix of \( E^{-1} \) restricted to the rows and columns labelled by the variables in \( x_1 \) and \( x_2 \) is \( B_{1,2}^{-1} \) and that labelled by the variables in \( x_{d-1} \) and \( x_d \) is \( B_{d-1,d}^{-1} \). For \( k \in [3, d-2] \) the submatrix restricted to the rows and columns labelled by \( x_{\sigma(k)} \) is \( B_{\sigma(k)}^{-1} \). We infer the following facts:

(I) The space spanned by the first \( u_1 - u_2 \) (that is \( w_1 \)) rows of \( V^{-1} \) is equal to the space spanned by the first \( w_1 \) rows of \( A \), the latter space is \( X_1 \).

(II) The space spanned by the first \( u_1 \) (that is \( w_1 + w_1w_2 \)) rows of \( V^{-1} \) is equal to the space spanned by the first \( w_1 + w_1w_2 \) rows of \( A \), the latter space is \( X_{1,2} \).

(III) The space spanned by the last \( u_d \) (that is \( w_{d-1} + w_{d-2}w_{d-1} \)) rows of \( V^{-1} \) is equal to the space spanned by the last \( w_{d-1} + w_{d-2}w_{d-1} \) rows of \( A \), the latter space is \( X_{d-1,d} \).
(IV) The space spanned by the last \( u_d - u_{d-1} \) (that is \( w_{d-1} \)) rows of \( V^{-1} \) is equal to the space spanned by the last \( w_{d-1} \) rows of \( A \), the latter space is \( X_d \).

(V) For \( k \in [3, d-2] \) the space spanned by the rows of \( V^{-1} \) that are numbered by \( t_k + 1 \) to \( t_k + u_k \) is equal to the space spanned by the rows of \( A \) labelled by \( x_{\sigma(k)} \), the latter space is \( X_{\sigma(k)} \).

\[ \Box \]

### 4.4.2 Reduction to almost set-multilinear ABP

**The outline:** Once the invariant spaces of \( g_f \) are computed, the reduction proceeds like this: As observed in the proof of Lemma 4.3, the matrix \( V \) in Algorithm 7 equals \( A^{-1}E \) where \( E \) looks as shown in Figure 4.4. If \( f = IMM_{w,d}(Ax) \) then \( f(Vx) = IMM_{w,d}(Ex) \). Owing to the structure of \( E \), \( f(Vx) \) is computed by a full-rank almost set-multilinear ABP (Definition 4.1), except that the ordering of the groups of variables occurring in the different layers of the ABP is unknown as \( \sigma \) is unknown. The ‘correct’ ordering along with a width vector can be retrieved by applying evaluation dimension, thereby completing the reduction. For a slightly neater presentation of the details (and with the intent of proving Lemma 4.2), we deviate from this strategy a little bit and make use of the layer spaces that have already been computed by Algorithm 7.

**The details:** Algorithm 7 computes the spaces \( y_1, y_{1,2}, y_3, \ldots, y_{d-2}, y_{d-1}, y_d \) which (according to Lemma 4.3) are either the spaces \( X_1, X_{1,2}, X_{\sigma(3)}, \ldots, X_{\sigma(d-2)}, X_{d-1}, X_d \) respectively, or the spaces \( X_d, X_{d-1}, X_{\sigma(3)}, \ldots, X_{\sigma(d-2)}, X_{1,2}, X_1 \) respectively, for some unknown permutation \( \sigma \) on \([3, d-2]\). The claim below shows how to correctly reorder these layer spaces.

**Claim 4.6** There is a randomized polynomial time algorithm that takes input the bases of the layer spaces \( y_1, y_{1,2}, y_3, \ldots, y_{d-2}, y_{d-1}, y_d \) and with probability at least \( 1 - n^{-\Omega(1)} \) reorders these layer spaces and outputs a width vector \( \omega \) such that the reordered sequence of spaces and \( \omega \) are:

1. either \( X_1, X_{1,2}, X_3, \ldots, X_{d-2}, X_{d-1}, X_d \) and \( (w_1, w_2, \ldots, w_{d-1}) \) respectively,

2. or \( X_d, X_{d-1}, X_{d-2}, \ldots, X_3, X_{1,2}, X_1 \) and \( (w_d, w_{d-1}, \ldots, w_1) \) respectively.

**Proof:** The algorithm employs evaluation dimension to uncover the permutation \( \sigma \). Assume that \( y_1, y_{1,2}, y_3, \ldots, y_{d-2}, y_{d-1}, y_d \) are the spaces \( X_1, X_{1,2}, X_{\sigma(3)}, \ldots, X_{\sigma(d-2)}, X_{d-1}, X_d \) respectively\(^1\). In this case, the algorithm reorders the spaces to a sequence \( X_1, X_{1,2}, X_3, \ldots, X_{d-2}, \ldots, X_{\sigma(d-2)}, X_{d-1}, X_d \).
\(X_{d-1,d}, X_d\) and outputs \(w' = w\). For every \(k \in [3, d-2]\), let \(z_k\) be a set of \(\dim(Y_k)\) many variables. Let \(z_1\) (similarly, \(z_d\)) be a set of \(\dim(Y_1)\) (similarly, \(\dim(Y_d)\)) variables, and let \(z_2\) (similarly, \(z_{d-1}\)) be a set of \(\dim(Y_{1,2}) - \dim(Y_1)\) (similarly, \(\dim(Y_{d-1,d}) - \dim(Y_d)\)) variables. Finally, let \(z = z_1 \uplus \ldots \uplus z_d\) be the set of these \(n\) fresh variables.

Compute a linear map \(\mu\) that maps \(x\) variables to linear forms in \(z\) variables such that the following conditions are satisfied:

(a) For every \(k \in [3, d-2]\), the linear forms corresponding\(^1\) to the basis vectors of \(Y_k\) map to distinct variables in \(z_k\).

(b) The linear forms corresponding to the basis vectors in \(Y_1\) (similarly, \(Y_d\)) map to distinct variables in \(z_1\) (similarly, \(z_d\)).

(c) The linear forms corresponding to the basis vectors in \(Y_{1,2}\) (similarly, \(Y_{d-1,d}\)) map to distinct variables in \(z_1 \uplus z_2\) (similarly, \(z_{d-1} \uplus z_d\)).

Conditions (b) and (c) can be simultaneously satisfied as the basis of \(Y_1\) (similarly, \(Y_d\)) is contained in the basis of \(Y_{1,2}\) (similarly, \(Y_{d-1,d}\)) by their very constructions in Algorithm 7. As \(f = \text{IMM}_{w,d}(Ax)\), the map \(\mu\) takes \(f\) to a polynomial \(h(z)\) that is computed by a full-rank ABP \(\Lambda'\) of width \(w\) and length \(d\) such that the sets of variables appearing in the \(d\) layers of \(\Lambda'\) from left to right are \(z_1, z_1 \uplus z_2, z_{\sigma^{-1}(3)}, \ldots, z_{\sigma^{-1}(d-2)}, z_{d-1} \uplus z_d, z_d\) in order.

The following observation, the proof of which is given later, helps find \(\sigma^{-1}\) incrementally from blackbox access to \(h(z)\). Let \(y_2 = z_1 \uplus z_2\) and \(y_j = z_1 \uplus z_2 \uplus z_{\sigma^{-1}(3)} \uplus \cdots \uplus z_{\sigma^{-1}(j)}\), for \(j \in [3, d-2]\).

**Observation 4.1** For every \(j \in [2, d-3]\) and \(k \in [3, d-2]\) such that \(z_k \not\subset y_j\),

1. \(\text{Evaldim}_{y_j \cup z_k}(h) < |z_k|\), if \(k = \sigma^{-1}(j + 1)\), and

2. \(\text{Evaldim}_{y_j \cup z_k}(h) > |z_k|\), if \(k \neq \sigma^{-1}(j + 1)\).

The proof of the observation also includes an efficient randomized procedure to compute \(\text{Evaldim}_{y_j \cup z_k}(h)\).

\(^1\)Recall, linear forms in \(x\) variables and vectors in \(\mathbb{R}^n\) are naturally identified with each other.
Finally, the algorithm outputs the reordered layer spaces \( y_1, y_{1,2}, y_{\sigma^{-1}(3)}, \ldots, y_{\sigma^{-1}(d-2)}, y_{d-1,d}, y_d \) which is the ordered sequence of spaces \( X_1, X_{1,2}, X_3, \ldots, X_{d-2}, X_{d-1,d}, X_d \). The width vector \( w' \) can be readily calculated now by inspecting the dimensions:

\[
\begin{align*}
 w'_1 &= \dim(X_1) = w_1, \\
 w'_2 &= \frac{\dim(X_{1,2}) - w_1}{w_1} = w_2, \\
 w'_k &= \frac{\dim(X_k)}{w_k-1} = w_k, \quad \text{for } k \in [3, d-2], \\
 w'_d &= \dim(X_d) = w_d, \quad \text{and} \\
 w'_{d-1} &= \frac{\dim(X_{d-1,d}) - w_d}{w_d} = w_{d-1}.
\end{align*}
\]

This gives \( w' = w \). This completes the proof the claim. Below we give the proof of Observation 4.1

**Proof of Observation 4.1:** Let \( Z_1 \cdot Z_2 \cdots Z_d \) be equal to \( A' \), the full-rank ABP of width \( w = (w_1, w_2, \ldots, w_{d-1}) \) computing \( h \), where the linear forms in \( Z_i \) are in \( z_{\sigma^{-1}(i)} \) variables for \( i \in [3, d-2] \), the linear forms in \( Z_1, Z_d \) are in variables \( z_1, z_d \) respectively, and the linear forms in \( Z_2, Z_{d-1} \) are in \( z_1 \uplus z_2, z_{d-1} \uplus z_d \) variables respectively.

Case 1: Suppose \( k = \sigma^{-1}(j + 1) \), implying \( |z_k| = w_jw_{j+1} \). Let \( G = Z_{j+2} \cdot Z_{j+3} \cdots Z_d \) and the \( t \)-th entry of \( G \) be \( g_t \) for \( t \in [w_{j+1}] \). As the linear forms in \( Z_1, Z_2, \ldots, Z_{j+1} \) are \( \mathbb{F} \)-linearly independent, for every \( t \in [w_{j+1}] \) there is a partial evaluation of \( h \) at \( y_j \uplus z_k \) variables that makes \( h \) equal to \( g_t \). Also, every partial evaluation of \( h \) at \( y_j \uplus z_k \) variables can be expressed as an \( \mathbb{F} \)-linear combination of \( g_1, g_2, \ldots, g_{w_{j+1}} \). Hence, from Observation 4.3 it follows, \( \text{Eval}_{y_j \uplus z_k}(h) = w_{j+1} < |z_k| \).

Case 2: Suppose \( k \neq \sigma^{-1}(j + 1) \). The variables \( z_k \) appear in the matrix \( Z_{\sigma(k)} \), so \( |z_k| = w_{\sigma(k)-1}w_{\sigma(k)} \). Let \( G = Z_{\sigma(k)+1} \cdot Z_{\sigma(k)+2} \cdots Z_d \) and the \( t \)-th entry of \( G \) be \( g_t \) for \( t \in [w_{\sigma(k)}] \). Further, let \( P = (p_{lm})_{l \in [w_j], m \in [w_{\sigma(k)-1}]} \) be equal to \( Z_{j+1} \cdot Z_{j+2} \cdots Z_{\sigma(k)-1} \). As the linear forms in \( Z_1, Z_2, \ldots, Z_j \) and \( Z_{\sigma(k)} \) are \( \mathbb{F} \)-linearly independent, there is a partial evaluation of \( h \) at the \( y_j \uplus z_k \) variables that makes \( h \) equal to \( p_{lm}g_t \) for \( l \in [w_j], m \in [w_{\sigma(k)-1}] \) and \( t \in [w_{\sigma(k)}] \). By Observation 4.3, \( \{g_t \mid t \in [w_{\sigma(k)}]\} \) are \( \mathbb{F} \)-linearly independent; using a proof similar to that of Observation 4.3 we can show that the polynomials \( \{p_{lm} \mid l \in [w_j], m \in [w_{\sigma(k)-1}]\} \) are also \( \mathbb{F} \)-linearly independent. This implies the set of polynomials \( \{p_{lm}g_t \mid l \in [w_j], m \in [w_{\sigma(k)-1}]\} \)
Let \( \text{Evaldim}_{y_j \uplus z_k}(h) = w_j w_{\sigma(k) - 1} w_{\sigma(k)} = w_j \cdot |z_k| > |z_k| \).

A randomized procedure to compute \( \text{Evaldim}_{y_j \uplus z_k}(h) \): Choose evaluation points \( a_1, \ldots, a_{n^2} \) for the variables \( y_j \uplus z_k \) independently and uniformly at random from a set \( S^{y_j \uplus z_k} \subset \mathbb{F}^{y_j \uplus z_k} \) with \( |S| = n^{O(1)} \). Output the dimension of the \( \mathbb{F} \)-linear space spanned by the polynomials \( h(a_1), \ldots, h(a_{n^2}) \) using Claim 2.3.

We argue that the above procedure outputs \( \text{Evaldim}_{y_j \uplus z_k}(h) \) with probability at least \( 1 - n^{-\Omega(1)} \). Let \( \text{Evaldim}_{y_j \uplus z_k}(h) = e \). Observe that in both Case 1 and 2, \( e \leq n^2 \). Also, in both the cases \( h \) can be expressed as

\[
h = \sum_{i \in [e]} f_i(y_j \uplus z_k) \cdot q_i,
\]

where \( f_i \) and \( q_i \) are variable disjoint. The polynomials \( q_1, \ldots, q_e \) are the polynomials \( g_1, \ldots, g_{w_j + 1} \) in Case 1; they are the polynomials \( \{p_{lm} g_t \mid l \in [w_j], m \in [w_{\sigma(k) - 1}] \text{ and } t \in [w_{\sigma(k)}]\} \) in Case 2. Just as we argue that \( q_1, \ldots, q_e \) are \( \mathbb{F} \)-linearly independent, we can show that \( f_1, \ldots, f_e \) are also \( \mathbb{F} \)-linearly independent. So, by Claim 2.3 the rank of the matrix \( M = (f_i(a_c))_{i,c \in [e]} \) is \( e \) with high probability. This implies the polynomials \( h(a_1), \ldots, h(a_e) \) are \( \mathbb{F} \)-linearly independent also with high probability. The correctness of the procedure follows from the observation that the dimension of the \( \mathbb{F} \)-linear space spanned by \( h(a_1), \ldots, h(a_{n^2}) \) is upper bounded by \( e \) (from Equation (4.6)).

\[\text{Note:}\] Until the algorithm in the claim is applied to reorder the spaces, Algorithm 4 is totally oblivious of the width vector \( w \) (it has been used only in the analysis thus far). So, due to the legitimacy of the transposition transformation mentioned at the beginning of this section, we may as well assume that the \( w' \) in the above claim is in fact our \( w \), and the output ordered sequence of spaces is \( X_1, X_{1,2}, X_3, \ldots, X_{d-2}, X_{d-1}, X_d \).

\[\text{Claim 4.7}\] Given bases of the spaces \( X_1, X_{1,2}, X_3, \ldots, X_{d-2}, X_{d-1}, X_d \) and \( w \), we can find an \( \hat{A} \in GL(n) \) in polynomial time such that \( f(\hat{A}x) \) is computable by a full-rank almost set-multilinear ABP of width \( w \).

\[\text{Proof:}\] Identify the variables \( x_1, \ldots, x_n \) with the variables \( x_1 \uplus \ldots \uplus x_d \) of IMM\(_{w,d} \) following the ordering prescribed in Section 2.3. The map \( x \mapsto \hat{A}x \) should satisfy the following conditions:
(a) For every $k \in [3, d - 2]$, the linear forms corresponding\(^1\) to the basis vectors of $X_k$ map to distinct variables in $x_k$.

(b) The linear forms corresponding to the basis vectors in $X_1$ (similarly, $X_d$) map to distinct variables in $x_1$ (similarly, $x_d$).

(c) The linear forms corresponding to the basis vectors in $X_{1,2}$ (similarly, $X_{d-1,d}$) map to distinct variables in $x_1 \cup x_2$ (similarly, $x_{d-1} \cup x_d$).

Conditions (b) and (c) can be simultaneously satisfied as the basis of $X_1$ (similarly, $X_d$) is contained in the basis of $X_{1,2}$ (similarly, $X_{d-1,d}$) by construction. Such an $\tilde{A}$ can be easily obtained.

We summarize the discussion in Algorithm 8.

### Algorithm 8 Reduction to full-rank almost set-multilinear ABP

**INPUT:** Bases of the layer spaces $Y_1, Y_{1,2}, Y_3, \ldots, Y_{d-2}, Y_{d-1,d}, Y_d$ from Algorithm 7.

**OUTPUT:** A $w \in \mathbb{N}^{d-1}$ and an $\tilde{A} \in \text{GL}(n)$ such that $f(\tilde{A}x)$ is computable by a full-rank almost set-multilinear ABP of width $w$.

1. Reorder the layer spaces to $X_1, X_{1,2}, X_3, \ldots, X_{d-2}, X_{d-1,d}, X_d$ and obtain $w$ (using Claim 4.6). /* This step succeeds with high probability if $f$ is equivalent to $\text{IMM}_{w,d}$ for some $w$. */

2. Find $\tilde{A} \in \text{GL}(n)$ from the reordered spaces and $w$ (using Claim 4.7).

**Comments on Algorithm 8:** The proof of Claim 4.6 includes Observation 4.1 which helps Algorithm 8 in step 1 to reorder the layer spaces. If $f$ is not equivalent to $\text{IMM}_{w,d}$ for some $w$ then Algorithm 8 may fail in step 1, as at some stage it may not be able to find a variable set $z_k$ such that $\text{Evaldim}_{y_j \cup z_k}(h) < |z_k|$ (see proof of Observation 4.1). When Algorithm 4 invokes Algorithm 8, if step 1 fails then the latter outputs 'Fail' and stops.

### 4.4.3 Reconstructing almost set-multilinear ABP

We prove Claim 4.1 in this section. Let $h = f(\tilde{A}x)$; identify $x$ with the variables $x_1 \cup \ldots \cup x_d$ of $\text{IMM}_{w,d}$ as before. From Claim 4.7, $h$ is computable by a full-rank almost set-multilinear ABP of width $w$. Algorithm 4 uses Algorithm 9 to reconstruct a full-rank almost set-multilinear ABP for $h$ and then it replaces $x$ by $\tilde{A}^{-1}x$ to output a full-rank ABP for $f$. The correctness of Algorithm 9 is presented as part of the proof of Claim 4.1. We begin with the following two observations.

\(^1\)Recall, linear forms in $x$ variables and vectors in $\mathbb{R}^n$ are naturally identified with each other.
Observation 4.2 If $h$ is computable by a full-rank almost set-multilinear ABP of width $w$ then there is a full-rank almost set-multilinear ABP of width $w$ in canonical form computing $h$.

Proof: Suppose $X_1 \cdot X_2 \cdots X_d$ is a full-rank almost set-multilinear ABP of width $w = (w_1, w_2, \ldots, w_{d-1})$ computing $h$. Let $X'_1 = (x_1^{(1)} \ x_2^{(1)} \ \cdots \ x_{w_1}^{(1)})$ and $X'_d = (x_1^{(d)} \ x_2^{(d)} \ \cdots \ x_{w_{d-1}}^{(d)})$. We show there are matrices $X'_2$ and $X'_{d-1}$ satisfying conditions (1b) and (2b) respectively of canonical form (defined in Section 4.2) such that $h = X'_1 \cdot X'_2 \cdot X_3 \cdots X_{d-2} \cdot X'_{d-1} \cdot X'_d$. We prove the existence of $X'_2 = (l'_{ij})_{i \in [w_1], j \in [w_2]}$; the proof for $X'_{d-1}$ is similar. It is sufficient to show that there is such an $X'_2$ satisfying $X_1 \cdot X_2 = X'_1 \cdot X'_2$. Denote the $j$-th entry of the $1 \times w_2$ matrix $X_1 \cdot X_2$ as $X_1 \cdot X_2(j)$. Similarly $X'_1 \cdot X'_2(j)$ represents the $j$-th entry of $X'_1 \cdot X'_2$. Let $g_j$ be the sum of all monomials in $X_1 \cdot X_2(j)$ of the following types: $x_i^{(1)} x_k^{(1)}$ for $k \in [i, w_1]$, and $x_i^{(1)} x_p^{(2)}$ for $p \in [w_1], q \in [w_2]$. Clearly, $X_1 \cdot X_2(j) = g_1 + g_2 + \cdots + g_{w_1}$. If $l''_{ij} \overset{\text{def}}{=} g_i/x_i^{(1)}$ then $X_1 \cdot X_2(j) = x_1^{(1)} l''_{1j} + x_2^{(1)} l''_{2j} + \cdots + x_{w_1}^{(1)} l''_{w_1,j}$. Since $l''_{ij}$ is the $(i, j)$-th entry of $X'_2$, we infer $X_1 \cdot X_2(j) = X'_1 \cdot X'_2(j)$. By definition, $x_k^{(1)}$ does not appear in $l''_{ij}$ for $k < i$, and thus condition (1b) is satisfied by $X'_2$. ■

Observation 4.3 Let $X_1 \cdot X_2 \cdots X_d$ be a full-rank almost set-multilinear ABP, and $C_k = X_k \cdots X_d$ for $k \in [2, d]$. Let the $l$-th entry of $C_k$ be $h_{kl}$ for $l \in [w_{k-1}]$. Then the polynomials $\{h_{k1}, h_{k2}, \ldots, h_{kw_{k-1}}\}$ are $\mathbb{F}$-linearly independent.

Proof: Suppose $\sum_{p=1}^{w_{k-1}} \alpha_p \cdot h_{kp} = 0$ such that $\alpha_p \in \mathbb{F}$ for $p \in [w_{k-1}]$, and not all $\alpha_p = 0$. Assume without loss of generality $\alpha_1 \neq 0$. Since the linear forms in $X_k, \ldots, X_d$ are $\mathbb{F}$-linearly independent, there is an evaluation of the variables in $x_k \uplus \cdots \uplus x_d$ to field constants such that $h_{k1} = 1$ and every other $h_{kp} = 0$ under this evaluation. This implies $\alpha_1 = 0$, contradicting our assumption. ■

Notations for Algorithm 9: For $k \in [d - 1]$, let $t_k = |x_1 \uplus x_2 \uplus \cdots \uplus x_k|$ and $m_k = |x_{k+1} \uplus x_{k+2} \uplus \cdots \uplus x_d|$. The $(i, j)$-th entry of a matrix $X$ is denoted by $X(i, j)$, and $e_{w_k,i}$ denotes a vector in $\mathbb{F}^{w_k}$ with the $i$-th entry 1 and other entries 0. Let $y_i$ denote the following partial assignment to the $x_i$ variables: $x_i^{(1)}, \ldots, x_i^{(t_k)}$ are kept intact, while the remaining variables are set to zero. Similarly, $z_j$ denotes the following partial assignment to the $x_d$ variables: $x_j^{(d)}, \ldots, x_{w_d}^{(d)}$ are kept intact, while the remaining variables are set to zero. The notation $h(a_i, x_j, b_j)$ means the variables $x_1 \uplus \cdots \uplus x_{k-1}$ are given the assignment $a_i \in \mathbb{F}^{t_{k-1}}$, and the variables $x_{k+1} \uplus \cdots \uplus x_d$ are given the assignment $b_j \in \mathbb{F}^{m_k}$. The connotations for
The function \( n^{O(1)} \) is a suitably large polynomial function in \( n \), say \( n^7 \).

**Proof of Claim 4.1:** By Observation 4.2, there is a full-rank ABP \( X'_1 \cdot X'_2 \cdots X'_d \) in canonical form computing \( h \). Hence \( X_1 = X'_1 = (x_1^{(1)} \ x_2^{(1)} \ldots \ x_{w_1}^{(1)}) \) and \( X_d = X'_d = (x_1^{(d)} \ x_2^{(d)} \ldots \ x_{w_d}^{(d)}) \).

We show next that with probability at least \( 1 - n^{-\Omega(1)} \), Algorithm 9 constructs \( X_2, X_3, \ldots, X_{d-1} \) such that \( X_2 = X'_2 \cdot T_2, \ X_{d-1} = T_{d-2}^{-1} \cdot X'_{d-1} \) and \( X_k = T_{k-1}^{-1} \cdot X'_k \cdot T_k \) for every \( k \in [3, d-2] \), where \( T_i \in \text{GL}(w_i) \) for \( i \in [2, d-2] \).

Steps 3–13: The matrix \( X_2 \) is formed in these steps. By Observation 4.3, the polynomials \( h_{31}, \ldots, h_{3w_2} \) are \( \mathbb{F} \)-linearly independent. Since \( b_1, b_2, \ldots, b_{w_2} \) are randomly chosen in step 3, the matrix \( T_2 \) with \((r, c)\)-th entry \( h_{3r}(b_c) \) is in \( \text{GL}(w_2) \) with high probability. Let \( X'_2 T_2(i, j) \) be the \((i, j)\)-th entry of \( X'_2 T_2 \). Observe that

\[
    h(y_1, x_2, b_j) = X'_2 T_2(i, j) \cdot x_i^{(1)} + \ldots + X'_2 T_2(w_1, j) \cdot x_{w_1}^{(1)}.
\]

As \( h(y_1, x_2, b_j) \) is a quadratic polynomial, we can compute it from blackbox access using the sparse polynomial interpolation algorithm in [KS01]. By induction on the rows, \( X_2(p, j) = X'_2 T_2(p, j) \) for every \( p \in [i + 1, w_1] \) and \( j \in [w_2] \). So in step 8, \( g_j = X'_2 T_2(i, j) \cdot x_i^{(1)} \) leading to \( X_2(i, j) = X'_2 T_2(i, j) \) in step 9.

Steps 15–23: The matrices \( X_3, \ldots, X_{d-2} \) are formed in these steps. By the time the algorithm reaches step 17, it has already computed \( X_2, \ldots, X_{k-1} \) such that \( X_2 = X'_2 T_2 \) and \( X_q = T_q^{-1} \cdot X'_q T_q \) for \( q \in [3, k-1] \), where \( T_q \in \text{GL}(w_q) \). So, \( X'_1 \cdots X'_{k-1} = X_1 \cdots X_{k-1} T_{k-1}^{-1} \). As the linear forms in \( X_1, \ldots, X_{k-1} \) are \( \mathbb{F} \)-linearly independent (otherwise the algorithm would have terminated in step 13 or 21), we can easily compute points \( \{a_1, a_2, \ldots, a_{w_{k-1}}\} \) satisfying the required condition in step 17. By Observation 4.3, the polynomials \( h_{(k+1)1}, \ldots, h_{(k+1)w_k} \) are \( \mathbb{F} \)-linearly independent. Since \( b_1, b_2, \ldots, b_{w_k} \) are randomly chosen in step 18, the matrix \( T_k \) with \((r, c)\)-th entry \( h_{(k+1)r}(b_c) \) is in \( \text{GL}(w_k) \) with high probability. Now observe that \( h(a_j, x_k, b_j) \) is the \((i, j)\)-th entry of \( T_{k-1}^{-1} \cdot X'_k T_k \), which implies \( X_k = T_{k-1}^{-1} \cdot X'_k T_k \) from step 20.

Steps 25–35: In these steps, matrix \( X_{d-1} \) is formed. The argument showing \( X_{d-1} = T_{d-2}^{-1} X'_{d-1} \) is similar to the argument used for steps 3–13, except that now we induct on columns instead of rows.
Algorithm 9 Reconstruction of full-rank almost set-multilinear ABP

**INPUT:** Blackbox access to an $n$ variate polynomial $h$ and the width vector $w$.

**OUTPUT:** A full-rank almost set-multilinear ABP of width $w$ in canonical form computing $h$.

1: Set $X_1 = (x_1^{(1)}, x_2^{(1)}, \ldots, x_{w_1}^{(1)})$ and $X_d = (x_1^{(d)}, x_2^{(d)}, \ldots, x_{w_{d-1}}^{(d)})^T$.

2: 

3: Choose $w_2$ random points $\{b_1, b_2, \ldots, b_{w_2}\}$ from $S^{w_2}$ such that $S \subseteq F$ and $|S| = n^{O(1)}$.

4: Set $i = w_1$.

5: while $i \geq 1$ do

6: for every $j \in [w_2]$ do

7: Interpolate the quadratic $h(y_i, x_2, b_j)$.

8: Set $g_j = h(y_i, x_2, b_j) - \sum_{p=i+1}^{w_1} X_2(p, j) \cdot x_1^{(1)}$.

9: If $g_j$ is not divisible by $x_1^{(1)}$, output ‘Fail’. Else, set $X_2(i, j) = g_j / x_1^{(1)}$.

10: end for

11: Set $i = i - 1$.

12: end while

13: If the linear forms in $X_2$ are not $F$-linearly independent, output ‘Fail’.

14: Set $k = 3$.

15: while $k \leq d - 2$ do

16: Find $w_{k-1}$ evaluations, $\{a_1, a_2, \ldots, a_{w_{k-1}}\} \subseteq F^{w_{k-1}}$, of $x_1 \cup x_2 \cup \cdots \cup x_{k-1}$ variables such that $X_1 \cdot X_2 \cdots X_{k-1}$ evaluated at $a_i$ equals $e_{w_{k-1}, i}$.

17: Choose $w_k$ random points $\{b_1, b_2, \ldots, b_{w_k}\}$ from $S^{w_k}$ such that $S \subseteq F$ and $|S| = n^{O(1)}$.

18: Interpolate the linear forms $h(a_i, x_k, b_j)$ for $i \in [w_{k-1}], j \in [w_k]$.

19: Set $X_k(i, j) = h(a_i, x_k, b_j)$ for $i \in [w_{k-1}], j \in [w_k]$.

20: If the linear forms in $X_k$ are not $F$-linearly independent, output ‘Fail’.

21: Set $k = k + 1$.

22: end while

23: 

24: Find $w_{d-2}$ evaluations, $\{a_1, a_2, \ldots, a_{w_{d-2}}\} \subseteq F^{w_{d-2}}$, of $x_1 \cup x_2 \cup \cdots \cup x_{d-2}$ variables such that $X_1 \cdot X_2 \cdots X_{d-2}$ evaluated at $a_i$ equals $e_{w_{d-2}, i}$.

25: Set $j = w_{d-1}$.

26: while $j \geq 1$ do

27: for every $i \in [w_{d-2}]$ do

28: Interpolate the quadratic $h(a_i, x_{d-1}, z_j)$.

29: Set $g_i = h(a_i, x_{d-1}, z_j) - \sum_{q=j+1}^{w_{d-1}} X_{d-1}(i, q) \cdot x_q^{(d)}$.

30: If $g_i$ is not divisible by $x_j^{(d)}$, output ‘Fail’. Else, set $X_{d-1}(i, j) = g_i / x_j^{(d)}$.

31: end for

32: Set $j = j - 1$.

33: end while

34: If the linear forms in $X_{d-1}$ are not $F$-linearly independent, output ‘Fail’.

35: Output $X_1 \cdot X_2 \cdots X_{d-1} \cdot X_d$ as the full-rank almost set-multilinear ABP for $h$. 

97
The output $\text{ABP } X_1 \ldots X_d$ is in canonical form as $X'_1 \ldots X'_d$ is also in canonical form. It is clear that the total running time of the algorithm is $(n, \beta)^{O(1)}$, where $\beta$ is the bit length of the coefficients of $h$ which influences the bit length of the values returned by the blackbox. \hfill \blacksquare
Chapter 5

Symmetries of IMM

In this chapter we determine $\mathcal{G}_{\text{IMM}}$ and show that IMM is characterized by $\mathcal{G}_{\text{IMM}}$. To the best of our knowledge, the group of symmetries of IMM$_{w,d}$ has not been studied previously. The contents of this chapter are from our work [KNST19].

The group of symmetries of an $n$ variate polynomial $f$, denoted as $\mathcal{G}_f$ is the set of all $A \in \text{GL}(n, \mathbb{F})$ such that $f(x) = f(Ax)$ (see Definition 2.16). Suppose for any $n$ variate degree $d$ polynomial $g$, $\mathcal{G}_g = \mathcal{G}_f$ implies $f = \alpha g$ for some non-zero $\alpha \in \mathbb{F}$. Then we say $f$ is characterized by $\mathcal{G}_f$. In this chapter, we determine the group of symmetries of the iterated matrix multiplication polynomial (IMM$_{w,d}$, as defined in Section 2.3) and show that IMM$_{w,d}$ is characterized by its group of symmetries.

5.1 The group $\mathcal{G}_{\text{IMM}}$

Before we begin, we make a note of a few notations and terminologies below.

- Calligraphic letters $\mathcal{H}$, $\mathcal{C}$, $\mathcal{M}$ and $\mathcal{I}$ denote subgroups of $\mathcal{G}_{\text{IMM}}$. Let $\mathcal{C}$ and $\mathcal{H}$ be subgroups of $\mathcal{G}_{\text{IMM}}$ such that $\mathcal{C} \cap \mathcal{H} = I_n$ and for every $H \in \mathcal{H}$ and $C \in \mathcal{C}$, $H \cdot C \cdot H^{-1} \in \mathcal{C}$. Then $\mathcal{C} \rtimes \mathcal{H}$ denotes the semidirect product of $\mathcal{C}$ and $\mathcal{H}$. The semidirect product of $\mathcal{C}$ and $\mathcal{H}$ is the set $\mathcal{C} \mathcal{H}$, which can be easily shown to be a subgroup of $\mathcal{G}_{\text{IMM}}$ and it also follows that $\mathcal{C}$ is a normal subgroup of $\mathcal{C} \rtimes \mathcal{H}$.

- For every $A \in \mathcal{G}_{\text{IMM}}$ the full-rank ABP obtained by replacing $x$ by $Ax$ in $Q_1 \cdot Q_2 \cdots Q_d$ is termed as the full-rank ABP determined by $A$. This full-rank ABP also computes IMM.

- Let $X$ be a matrix with entries as linear forms in $y \cup z$ variables. We break $X$ into two parts $X(y)$ and $X(z)$ such that $X = X(y) + X(z)$. The $(i,j)$-th linear form in $X(y)$
(respectively \(X(z)\)) is the part of the \((i, j)\)-th linear form of \(X\) in \(y\) (respectively \(z\)) variables.

**Three subgroups of \(G_{\text{IMM}}\):** As before, let \(w = (w_1, w_2, \ldots, w_{d-1})\) and \(w_k > 1\) for every \(k \in [d-1]\). In Theorem 5.1 below, we show that \(G_{\text{IMM}}\) is generated by three special subgroups.

1. **Transposition subgroup \(\mathcal{T}\):** If \(w_k \neq w_{d-k}\) for any \(k \in [d-1]\) then \(\mathcal{T}\) is the trivial group containing only \(I_n\). Otherwise, if \(w_k = w_{d-k}\) for every \(k \in [d-1]\) then \(\mathcal{T}\) is the group consisting of two elements \(I_n\) and \(T\). The matrix \(T\) is such that the full-rank ABP determined by \(T\) is \(Q^T_d \cdot Q^T_{d-1} \cdots Q^T_1\). Clearly, \(T\) is a permutation matrix and \(T^2 = I_n\).

2. **Left-right multiplications subgroup \(M\):** An \(M \in \text{GL}(n)\) is in \(M\) if and only if the full-rank ABP \(X_1 \cdot X_2 \cdots X_d\) determined by \(M\) has the following structure: There are matrices \(A_1, \ldots, A_{d-1}\) with \(A_k \in \text{GL}(w_k)\) for every \(k \in [d-1]\), such that \(X_1 = Q_1 \cdot A_1, X_d = A_{d-1}^{-1} \cdot Q_d,\) and \(X_k = A_{k-1}^{-1} \cdot Q_k \cdot A_k\) for \(k \in [2, d-1]\). It is easy to verify that \(M\) is a subgroup of \(G_{\text{IMM}}\) and is isomorphic to the direct product \(\text{GL}(w_1) \times \cdots \times \text{GL}(w_{d-1})\).

3. **Corner translations subgroup \(C\):** A matrix \(C \in \text{GL}(n)\) is in \(C\) if and only if the full-rank ABP \(X_1 \cdot X_2 \cdots X_d\) determined by \(C\) has the following structure: There are two sets \(\{C_{11}, C_{12}, \ldots, C_{1w_2}\}\) and \(\{C_{d1}, C_{d2}, \ldots, C_{dw_{d-2}}\}\) containing anti-symmetric matrices in \(\mathbb{F}^{w_1 \times w_1}\) and \(\mathbb{F}^{w_{d-1} \times w_{d-1}}\) respectively such that \(X_2 = Q_2 + Y_2\) and \(X_{d-1} = Q_{d-1} + Y_{d-1}\), where \(Y_2 \in \mathbb{F}[x_1]^{w_1 \times w_2}\) (respectively \(Y_{d-1} \in \mathbb{F}[x_d]^{w_{d-2} \times w_{d-1}}\)) is a matrix with its \(i\)-th column (respectively \(i\)-th row) equal to \(C_{1i} \cdot Q^T_1\) (respectively \(Q^T_d \cdot C_{di}\)). For every other \(k \in [d] \setminus \{2, d-1\}\), \(X_k = Q_k\). Observe that \(Q_1 \cdot C_{1i} \cdot Q^T_1 = Q^T_d \cdot C_{di} \cdot Q_d = 0\). It can also be verified that \(C\) is an abelian subgroup of \(G_{\text{IMM}}\) and is isomorphic to the direct product \(\alpha_{w_1} \times \alpha_{w_{d-2}}\), where \(\alpha_w\) is the group of \(w \times w\) anti-symmetric matrices under matrix addition and \(\alpha_{w_k}^k\) is the \(k\) times direct product of this group.

**Theorem 5.1 (Symmetries of IMM)** \(G_{\text{IMM}} = C \rtimes H\), where \(H = M \rtimes T\).

We prove Theorem 5.1 in Section 5.2. Following are a couple of remarks on it.

(a) **Characterization:** We prove IMM is characterized by \(G_{\text{IMM}}\) in Lemma 5.1. The groups \(M\) and \(C\) generate the ‘continuous symmetries’ of IMM.

(b) **Comparison with a related work:** In [Ges16] a different choice of the IMM polynomial is considered, namely the trace of a product of \(d\) square symbolic matrices – let us call this
polynomial IMM’. The group of symmetries of IMM’ is determined in [Ges16] and it is shown that IMM’ is characterized by \( G_{\text{imm}}’ \). The group of symmetries of IMM’, like IMM, is generated by the transposition subgroup, the left-right multiplication subgroup, and (instead of the corner translations subgroup) the *circular transformations subgroup* – an element in this subgroup cyclically rotates the order of the matrices and hence does not change the trace of the product.

### 5.2 Proof of Theorem 5.1

We recall the notations introduced in the previous chapter. Let \( X_1 \cdot X_2 \cdots X_d \) be a full-rank ABP with linear forms as entries in matrices \( X_1, \ldots, X_d \). Then \( X_k \) denotes the space spanned by the linear forms in \( X_k \) for \( k \in \{1, 2, \ldots, d\} \). \( X_1 \cdot X_2 \cdot X_3 \cdot \ldots \cdot X_d \) denotes the space spanned by the linear forms in \( X_1, X_2, \) and \( X_{d-1}, X_d \) respectively. Similarly from Section 2.3, \( Q_1 \cdots Q_d \) is the ABP computing \( f \), and \( Q_k \) denotes the space spanned by the linear forms in \( Q_k \) for \( k \in \{1, 2, \ldots, d\} \). \( Q_{d-1,d} \) denotes the space spanned by the linear forms in \( Q_1, Q_2, \) and \( Q_{d-1}, Q_d \) respectively. We begin with the following observation which is immediate from Lemma 4.2.

**Observation 5.1** If \( X_1 \cdot X_2 \cdots X_d \) is a width \( w' = (w'_1, w'_2, \ldots, w'_{d-1}) \) full-rank ABP computing \( \text{IMM}_{w,d} \) then either

1. \( w'_k = w_k \) for \( k \in \{d-1\} \), and the spaces \( X_1, X_1, 2, \ldots, X_d \) are the spaces \( Q_1, Q_{1, 2}, Q_3, \ldots, Q_{d-1, d}, Q_d \) respectively, or

2. \( w'_k = w_{d-k} \) for \( k \in \{d-1\} \), and the spaces \( X_1, X_1, 2, \ldots, X_d \) are the spaces \( Q_d, Q_{d-1, d}, Q_{d-2}, \ldots, Q_{1, 2}, Q_1 \) respectively.

From the definitions of \( T, M \) and \( C \) it follows that \( C \cap M = C \cap I = M \cap I = I_n \). The claim below shows \( G_{\text{imm}}' \) is generated by \( C, M \) and \( T \).

**Claim 5.1** For every \( A \in G_{\text{imm}}' \), there exist \( C \in C, M \in M \) and \( \bar{T} \in T \) such that \( A = C \cdot M \cdot \bar{T} \).

**Proof:** Let \( X_1 \cdot X_2 \cdots X_d \) be the full-rank ABP \( A \) of width \( w \) determined by \( A \). If \( w_k = w_{d-k} \) for \( k \in \{d-1\} \) then the spaces \( X_1, X_1, 2, \ldots, X_d \) are either equal to \( Q_1, Q_{1, 2}, Q_3, \ldots, Q_{d-1, d}, Q_d \) respectively or \( Q_d, Q_{d-1, d}, Q_{d-2}, \ldots, Q_{1, 2}, Q_1 \) respectively (from Observation 5.1). Otherwise if \( w_k \neq w_{d-k} \) for any \( k \in \{d-1\} \) then the spaces \( X_1, X_1, 2, \ldots, X_d \) have only one choice and are equal to \( Q_1, Q_{1, 2}, Q_3, \ldots, Q_{d-1, d}, Q_d \) respectively. We deal with these two choices of layer spaces separately.

---

1The complexities of IMM and IMM’ are polynomially related to each other, in particular both are complete for algebraic branching programs under p-projections. But their groups of symmetries are slightly different.
Case A: Suppose $X_1, X_{1,2}, X_3, \ldots, X_{d-1}, d, X_d$ are equal to $Q_1, Q_{1,2}, Q_3, \ldots, Q_{d-1}, d, Q_d$ respectively. Hence $A$ looks as shown in Figure 5.1. The linear forms in $X_2, X_{d-1}$ are in variables $x_1 \uplus x_2 \uplus x_3 \uplus \ldots \uplus x_d$.

Figure 5.1: Matrix $A$ in $G_{\text{IMM}}$.

Since $A$ is a full-rank ABP and each monomial in IMM contains one variable from each set $x_k$,

$$
X_1 \cdot X_2(x_2) \cdot \left(\prod_{k=3}^{d-2} X_k\right) \cdot X_{d-1} \cdot X_d = \text{IMM},
$$

and also

$$
X_1 \cdot X_2(x_2) \cdot \prod_{k=3}^{d-2} X_k \cdot X_{d-1} \cdot X_d = 0 \quad \text{and} \quad X_1 \cdot X_2(x_2) \cdot \prod_{k=3}^{d-2} X_k \cdot X_{d-1} \cdot X_d = 0
$$

implying

$$
X_1 \cdot X_2(x_1) = 0^T_{w_2} \quad \text{and} \quad X_{d-1} \cdot X_d = 0_{w_{d-2}},
$$

where $0_w$ is a zero (column) vector in $F^w$. Observation 5.2 proves the existence of a matrix $M \in \mathcal{M}$ such that the full-rank ABP determined by $M$ is $X_1 \cdot X_2(x_2) \cdot X_3 \cdot \ldots \cdot X_{d-2} \cdot X_{d-1} \cdot X_d$.

**Observation 5.2** There are matrices $A_1, \ldots, A_{d-1}$ with $A_k \in \text{GL}(w_k)$ for every $k \in [d-1]$, such that $X_1 = Q_1 \cdot A_1, X_2(x_2) = A_1^{-1} \cdot Q_2 \cdot A_2, X_{d-1}(x_{d-1}) = A_{d-2}^{-1} \cdot Q_{d-1} \cdot A_{d-1}, X_d = A_{d-1}^{-1} \cdot Q_d$, and $X_k = A_k^{-1} \cdot Q_k \cdot A_k$ for $k \in [3, d-2]$.

1We abuse notation slightly and write the $1 \times 1$ matrix $[\text{IMM}]_{1 \times 1}$ as IMM.
Proof: To simplify notations, we write $X_2(x_2), X_{d-1}(x_{d-1})$ as $X_2, X_{d-1}$ respectively. We have

$$X_1 \cdot X_2 \cdots X_{d-1} \cdot X_d = Q_1 \cdot Q_2 \cdots Q_{d-1} \cdot Q_d = \text{IMM},$$

where the dimensions of the matrices $X_k$ and $Q_k$ are the same, and the set of variables appearing in both $X_k$ and $Q_k$ is $x_k$, for every $k \in [d]$. Since the linear forms in $X_1$ are $\mathbb{F}$-linearly independent, there is an $A_1 \in \text{GL}(w_1)$ such that $X_1 = Q_1 \cdot A_1$, implying

$$Q_1 \cdot [A_1 \cdot X_2 \cdots X_{d-1} \cdot X_d - Q_2 \cdots Q_{d-1} \cdot Q_d] = 0$$

$$\Rightarrow \quad X_2 \cdots X_{d-1} \cdot X_d = A_1^{-1} \cdot Q_2 \cdots Q_{d-1} \cdot Q_d,$$

as the formal variable entries of $Q_i$ do not appear in the matrices $X_k, Q_k$ for $k \in [2, d]$. The rest of the proof proceeds inductively: Suppose for some $k \in [2, d-1],$

$$X_k \cdots X_{d-1} \cdot X_d = A_{k-1}^{-1} \cdot Q_k \cdots Q_{d-1} \cdot Q_d, \quad \text{where } A_{k-1} \in \text{GL}(w_{k-1}).$$

Let $p_k = \sum_{i=k+1}^{d} |x_i|$. Since the linear forms in $X_{k+1}, \ldots, X_{d-1}, X_d$ are $\mathbb{F}$-linearly independent, for every $l \in [w_k]$ there is a point $a_l \in \mathbb{F}^n$ such that the $w_k \times 1$ matrix $X_{k+1} \cdots X_{d-1} \cdot X_d$ evaluated at $a_l$ has 1 at the $l$-th position and all its other entries are zero. Let $A_k$ be the $w_k \times w_k$ matrix such that the $l$-th column of $A_k$ is equal to $Q_{k+1} \cdots Q_{d-1} \cdot Q_d$ evaluated at $a_l$. Then, $X_k = A_{k-1}^{-1} \cdot Q_k \cdot A_k$. As the linear forms in $X_k$ and $Q_k$ are $\mathbb{F}$-linearly independent, it must be that $A_k \in \text{GL}(w_k)$. Putting this expression for $X_k$ in the equation above and arguing as before, we get a similar equation with $k$ replaced by $k+1$. The proof then follows by induction. $\blacksquare$

We now show the existence of a $C \in \mathcal{C}$ such that the full-rank ABP determined by $C \cdot M$ is $X_1 \cdot X_2 \cdots X_d$, from which the claim follows by letting $T = I_n$. Since the linear forms in $X_1$ are $\mathbb{F}$-linearly independent, there are $w_1 \times w_1$ matrices $\{C_{11}, C_{12}, \ldots, C_{1w_2}\}$ such that the $i$-th column of $X_2(x_1)$ is $C_{1i} X_1^T$. So from Equation (5.1), $X_1 \cdot C_{1i} \cdot X_1^T = 0$ (equivalently $Q_1 \cdot C_{1i} \cdot Q_1^T = 0$) implying $C_{1i}$ is an anti-symmetric matrix for every $i \in [w_2]$. Similarly, there are $w_{d-1} \times w_{d-1}$ anti-symmetric matrices $\{C_{d1}, C_{d2}, \ldots, C_{dw_{d-2}}\}$ such that the $i$-th row of $X_{d-1}(x_d)$ is $X_d^T C_{di}$. Let $C \in \text{GL}(n)$ be such that the ABP determined by it is $Q_1 Q_2 Q_3 \cdots Q_{d-2} Q'_{d-1} Q_d$ where $Q'_2 = Q_2 + Y_2$ and $Q'_{d-1} = Q_{d-1} + Y_{d-1}$, the $i$-th column (respectively $i$-th row) of $Y_2$ (respectively $Y_{d-1}$) is $C_{1i} Q_1^T$ (respectively $Q_{d-1}^T C_{di}$). By construction, $C \in \mathcal{C}$ and the ABP determined by $C \cdot M$ is $X_1 \cdot X_2 \cdots X_d$.

Case B: Suppose $X_1, X_{1,2}, X_3, \ldots, X_{d-1,d}, X_d$ are the spaces $Q_d, Q_{d-1,d}, Q_{d-2}, \ldots, Q_{1,2}, Q_1$ respectively.
tively. This implies \( w_k = w_{d-k} \) for \( k \in [d-1] \) and hence the full-rank ABP determined by \( T \) is \( Q_d^T \cdot Q_{d-1}^T \cdots Q_1^T \). From here the existence of \( M \in \mathcal{M} \) and \( C \in \mathcal{C} \) such that the full-rank ABP determined by \( M \cdot C \cdot T \) is \( X_1 \cdot X_2 \cdots X_d \) follows just as in Case A. This completes the proof of the claim.

Observe that if \( T \in \mathcal{I} \) then for every \( M \in \mathcal{M} \), \( T \cdot M \cdot T^{-1} \in \mathcal{M} \). Let \( \mathcal{H} = \mathcal{M} \times \mathcal{I} \). Clearly, \( \mathcal{C} \cap \mathcal{H} = I_n \). The following claim along with Claim 5.1 then conclude the proof of Theorem 5.1.

**Claim 5.2** For every \( C \in \mathcal{C} \) and \( H \in \mathcal{H} \), \( H \cdot C \cdot H^{-1} \in \mathcal{C} \).

**Proof:** Let \( H = M \cdot T \) where \( M \in \mathcal{M} \) and \( T \in \mathcal{I} \), and \( A = MT \cdot C \cdot T^{-1}M^{-1} \). Suppose \( X_1 \cdot X_2 \cdots X_{d-1} \cdot X_d \) is the ABP determined by \( A \). The matrices \( T \) and \( T^{-1} \) in \( A \) together ensure that the spaces \( X_1, X_1, X_2, X_3, \ldots, X_{d-1}, X_d \) are equal to \( Q_1, Q_1, Q_2, Q_3, \ldots, Q_{d-1}, Q_d \) respectively. Also the matrices \( M \) and \( M^{-1} \) together ensure that \( X_i = Q_i \) for \( i \in [d] \setminus \{2, d-1\} \), \( X_2(x_2) = Q_2 \) and \( X_{d-1}(x_{d-1}) = Q_{d-1} \). Arguing as in Claim 5.1, we can infer that \( A \in \mathcal{C} \).

### 5.3 Characterization of IMM by \( \mathcal{G}_{\text{IMM}} \)

For every \( f = \alpha \cdot \text{IMM} \), where \( \alpha \in \mathbb{F} \) and \( \alpha \neq 0 \), it is easily observed that \( \mathcal{G}_f = \mathcal{G}_{\text{IMM}} \). We prove the converse in the following lemma for any homogeneous degree \( d \) polynomial in the \( x \) variables.

**Lemma 5.1** Let \( f \) be a homogeneous degree \( d \) polynomial in \( n \) variables \( x = x_1 \uplus \ldots \uplus x_d \). If \( |\mathbb{F}| > d + 1 \) and the left-right multiplications subgroup \( \mathcal{M} \) of \( \mathcal{G}_{\text{IMM}} \) is contained in \( \mathcal{G}_f \) then \( f = \alpha \cdot \text{IMM} \) for some nonzero \( \alpha \in \mathbb{F} \).

**Proof:** First, we show that such an \( f \) is set-multilinear in the sets \( x_1, \ldots, x_d \). Every monomial in \( f \) has exactly one variable from each of the sets \( x_1, \ldots, x_d \). As \( |\mathbb{F}| > d + 1 \), there is a nonzero \( \rho \in \mathbb{F} \) that is not an \( e \)-th root of unity for any \( e \leq d \). Every element in \( \mathcal{M} \) is defined by \( d - 1 \) matrices \( A_1, \ldots, A_{d-1} \) such that \( A_k \in \text{GL}(w_k) \) for every \( k \in [d - 1] \). For a \( k \in [d - 1] \), consider the element \( M \in \mathcal{M} \) that is defined by \( A_k = \rho \cdot I_{w_k} \) and \( A_l = I_{w_l} \) for \( l \in [d - 1] \) and \( l \neq k \). Then, \( f(M \cdot x) = f(x_1, \ldots, \rho x_k, \rho^{-1} x_{k+1}, \ldots, x_d) \), which by assumption is \( f \). Comparing the coefficients of the monomials of \( f(M \cdot x) \) and \( f \), we observe that in every monomial of \( f \) the number of variables from \( x_k \) and \( x_{k+1} \) must be the same as \( \rho \) is not an \( e \)-th root of unity for any \( e \leq d \). Since \( k \) is chosen arbitrarily and \( f \) is homogeneous of degree \( d \), \( f \) must be set-multilinear in the sets \( x_1, \ldots, x_d \).
The proof is by induction on the degree of $f$. For $i \in [w_1]$, let $x_{2i}$ be the set of variables in the $i$-th row of $Q_2$, and $Q_{2i}$ be the $1 \times w_2$ matrix containing the $i$-th row of $Q_2$. Let $\text{IMM}_i$ be the degree $d - 1$ iterated matrix multiplication polynomial computed by the ABP $Q_{2i} \cdot Q_3 \cdots Q_d$. As $f$ is set-multilinear, it can be expressed as

$$f = g_1 \cdot x^{(1)}_1 + \ldots + g_{w_1} \cdot x^{(1)}_{w_1},$$

(5.2)

where $g_1, \ldots, g_{w_1}$ are set-multilinear polynomials in the sets $x_2, \ldots, x_d$. Moreover, we can argue that $g_i$ is set-multilinear in $x_{2i}, x_3, \ldots, x_d$ as follows: Consider an $N \in \mathcal{M}$ that is defined by a diagonal matrix $A_1 \in \text{GL}(w_1)$ whose $(i,i)$-th entry is $\rho$ and all other diagonal entries are 1; every other $A_l = I_{w_l}$ for $l \in [2, d - 1]$. The transformation $N$ scales the variable $x^{(1)}_i$ by $\rho$ and the variables in $x_{2i}$ by $\rho^{-1}$. By comparing the coefficients of the monomials of $f(N \cdot x)$ and $f$, we can conclude that $g_i$ is set-multilinear in $x_{2i}, x_3, \ldots, x_d$ for every $i \in [w_1]$.

Let $\mathcal{M}'$ be the subgroup of $\mathcal{M}$ containing those $M \in \mathcal{M}$ for which $A_1 = I_{w_1}$. From Equation (5.2), we can infer that $g_i(M \cdot x) = g_i$ for $M \in \mathcal{M}'$, and hence the left-right multiplications subgroup of $\mathcal{G}_{\text{IMM}_i}$ is contained in the group of symmetries of $g_i$. As degree of $g_i$ is $d - 1$, by induction $g_i = \alpha_i \cdot \text{IMM}_i$ for some nonzero $\alpha_i \in \mathbb{F}$ and

$$f = \alpha_1 \cdot \text{IMM}_1 \cdot x^{(1)}_1 + \ldots + \alpha_{w_1} \cdot \text{IMM}_{w_1} \cdot x^{(1)}_{w_1},$$

(5.3)

Next we show that $\alpha_1 = \ldots = \alpha_{w_1}$ thereby completing the proof.

For an $i \in [2, w_1]$, let $A_1 \in \text{GL}(w_1)$ be the upper triangular matrix whose diagonal entries are 1, the $(1,i)$-th entry is 1 and remaining entries are zero. Let $U$ be the matrix in $\mathcal{M}$ defined by $A_1$ and $A_l = I_{w_l}$ for $l \in [2, d - 1]$. The transformation $U$ has the following effect on the variables:

$$x^{(1)}_i \mapsto x^{(1)}_1 + x^{(1)}_i \quad \text{and}$$

$$x^{(2)}_{1j} \mapsto x^{(2)}_{1j} - x^{(2)}_{ij} \quad \text{for every } j \in [w_2],$$

every other $x$ variable maps to itself. Applying $U$ to $f$ in Equation (5.3) we get

$$f = f(U \cdot x)$$

$^1$The base case $d = 1$ is trivial to show.
\[
\alpha_1 \cdot (\text{IMM}_1 - \text{IMM}_i) \cdot x_1^{(1)} + \ldots + \alpha_i \cdot \text{IMM}_i \cdot (x_i^{(1)} + x_i^{(1)}) + \ldots + \alpha_{w_1} \cdot \text{IMM}_{w_1} \cdot x_{w_1}^{(1)}
\]
\[
= f + (\alpha_i - \alpha_1) \cdot \text{IMM}_i \cdot x_1^{(1)},
\]
\[
\Rightarrow \alpha_i - \alpha_1 = 0.
\]

Since this is true for any \(i \in [2, w_1]\), we have \(\alpha_1 = \ldots = \alpha_{w_1}\).
Chapter 6

Average-case LMF and reconstruction of low width ABP

In this chapter we present our algorithms for average-case LMF and average-case ABP reconstruction. The contents of this chapter are from [KNS19].

6.1 Introduction

Algebraic branching program (ABP) (see Definition 2.4) is a powerful circuit model that captures the complexity of computing polynomials like the determinant and the IMM. Two related problems average-case linear matrix factorization (LMF) (Problem 1.1) and average-case ABP reconstruction (Problem 1.2) were defined in Section 1.2.2. The average-case LMF is an easier problem than average-case ABP reconstruction, as in the former we have blackbox access to \( w^2 \) correlated entries of a random \( (w,d,n) \)-matrix product, whereas in the latter we have blackbox access to just one polynomial computed by a random \( (w,d,n) \)-ABP. We give an efficient randomized algorithm for average-case LMF over \( \mathbb{F}_q \) in Theorem 1.2, when \( n \geq 2w^2 \). The algorithm for average-case LMF in Theorem 1.2 also helps us to give an algorithm for average-case ABP reconstruction over \( \mathbb{F}_q \) in Theorem 1.3. We restate both the theorems below for the convenience of the reader. Throughout this chapter we assume \( \mathbb{F} = \mathbb{F}_q \) with \( \text{char}(\mathbb{F}_q) \geq (dn)^7 \) and \( \text{char}(\mathbb{F}_q) \nmid w \) (see the first remark after Theorem 1.2 in Section 1.2.2), and \( \mathbb{L} \) the extension field \( \mathbb{F}_{q^w} \).

**Theorem 1.2 (Average-case LMF)** For \( n \geq 2w^2 \), there is a randomized algorithm that takes as input blackbox access to \( w^2 \), \( n \) variate, degree \( d \) polynomials \( \{f_{st}\}_{s,t \in [w]} \) that constitute the entries of a random \( (w,d,n) \)-matrix product \( F = X_1 \cdot X_2 \cdots X_d \) over \( \mathbb{F} \), and with probability
1 − \frac{(wd)^{O(1)}}{q} \quad \text{returns } w \times w \text{ linear matrices } Y_1, Y_2, \ldots, Y_d \text{ over } \mathbb{L} \text{ satisfying } F = \prod_{i=1}^{d} Y_i. \text{ The algorithm runs in } (dn \log q)^{O(1)} \text{ time and queries the blackbox at points in } \mathbb{L}^n.

**Theorem 1.3 (Average-case ABP reconstruction)** For \( n \geq 4w^2 \) and \( d \geq 5 \), there is a randomised algorithm that takes as input blackbox access to an \( n \) variate, degree \( d \) polynomial \( f \) computed by a random \((w, d, n)\)-ABP over \( \mathbb{F} \), and with probability \( 1 - \frac{(wd)^{O(1)}}{q} \) \quad \text{returns a } (w, d, n)\-ABP over \( \mathbb{L} \) computing \( f \). The algorithm runs in time \( (d^w n \log q)^{O(1)} \) and queries the blackbox at points in \( \mathbb{L}^n \).

Detailed remarks on both the above Theorems are given in Section 1.2.2. Theorem 1.2 gives a worst-case reconstruction algorithm for pure matrix products (see the third remark after Theorem 1.2 in Section 1.2.2). Similarly Theorem 1.3 gives worst-case reconstruction for an ABP satisfying a set of non-degeneracy conditions which we state in Section 6.4.3. The algorithms in the above theorems are given in Section 6.1.1.

The proof of Theorem 1.2 requires an efficient equivalence test for the determinant. Recall, an \( n \) variate polynomial \( f(x) \) is an affine projection via a full-rank transformation of \( \text{Det}_w \), if there is an \( A \in \mathbb{F}^{w^2 \times n} \) of rank \( w^2 \) and an \( a \in \mathbb{F}^{w^2} \) such that \( f = g(Ax + a) \). Further, for \( w^2 = n \), \( f \) is equivalent to \( g \) if there is an \( A \in \text{GL}(n, \mathbb{F}) \) such that \( f = g(Ax) \). Given blackbox access to a \( n \) variate, degree \( w \) polynomial \( f \), where \( n \geq w^2 \), the theorem below below determines whether \( f \) is an affine projection of \( \text{Det}_w \) via a full-rank transformation over finite fields – it returns a \( B \in \mathbb{L}^{w^2 \times n} \) of rank \( w^2 \) and a \( b \in \mathbb{L}^{w^2} \).

**Theorem 6.1** There is a randomized algorithm that takes as input blackbox access to a \( n \) variate, degree \( w \) polynomial \( f \in \mathbb{F}[x] \), where \( n \geq w^2 \), and does the following with probability \( 1 - \frac{n^{O(1)}}{q} \). If \( f \) is affine equivalent to \( \text{Det}_w \) then it outputs a \( B \in \mathbb{L}^{w^2 \times n} \) of rank \( w^2 \) and a \( b \in \mathbb{L}^{w^2} \) such that \( f = \text{Det}_w(Bx + b) \), else it outputs ‘no such \( B \) and \( b \) exists’. The algorithm runs in \( (n \log q)^{O(1)} \) time and queries the blackbox at points in \( \mathbb{L}^n \).

**Remarks:** Similar to the case of \( \text{IMM}_{w,d} \), using variable reduction (Algorithm 1) and translation equivalence testing (Algorithm 2) the above problem is reduced to equivalence testing of \( \text{Det}_w \). Below we compare our algorithm with the algorithms for equivalence testing of \( \text{Det}_w \) in [Kay12a] and [GGKS19].

1. **Comparison to [Kay12a]:** An efficient equivalence test for the determinant over \( \mathbb{C} \) was given in [Kay12a]. The computation model in [Kay12a] assumes that arithmetic over \( \mathbb{C} \)

\[ q \geq (dn)^7 \text{ the probability is } 1 - \frac{1}{(dn)^{100}}. \]

\[ q \geq (dn)^7 \text{ the probability is } 1 - \frac{1}{(dn)^{100}}. \]
and root finding of univariate polynomials over \( \mathbb{C} \) can be done efficiently. While we follow the general strategy of analyzing the Lie algebra of the determinant and reduction to PS-equivalence from [Kay12a], our algorithm is somewhat simpler: Unlike [Kay12a], our algorithm does not involve the Cartan subalgebras and is almost the same as the simpler equivalence test for the permanent polynomial in [Kay12a]. The simplification is achieved by showing that the characteristic polynomial of a random element of the Lie algebra of \( \text{Det}_w \) splits completely over \( \mathbb{L} \) with high probability (Lemma 6.5) – this is crucial for Theorem 1.2 as it allows the algorithm to output a matrix factorization over a fixed low extension of \( \mathbb{F} \), namely \( \mathbb{L} \).

2. **Comparison to [GGKS19]:** After this work, [GGKS19] gave an algorithm for determinant equivalence test over finite fields\(^1\) where the output of the algorithm is over the base field itself instead of an extension field like in our case. Their work also gave a determinant equivalence test over \( \mathbb{Q} \) where the output is over a degree \( w \) extension of \( \mathbb{Q} \). The algorithm in [GGKS19] is different from ours and is achieved by giving a polynomial time randomized reduction from the equivalence testing problem to a well-studied problem in symbolic computation which they call the ‘full matrix algebra isomorphism’ problem [Rón87, Rón90, BR90, IRS12].

### 6.1.1 Algorithms and their analysis

The algorithms mentioned in Theorem 1.2, 1.3 and 6.1 are given in Algorithm 10, 11 and 12 respectively. In this section, we briefly discuss their correctness and complexity – for the missing details, we allude to the relevant parts of the subsequent sections.

#### 6.1.1.1 Analysis of Algorithm 10

Since \( F = X_1 \cdot X_2 \cdots X_d \) is a random \( (w, d, n) \)-matrix product, with probability \( 1 - (wn)^{-\Omega(1)} \), the first two properties of a pure product are satisfied: Every \( X_i \) is a full-rank linear matrix, and \( \det(X_1), \det(X_2), \ldots, \det(X_d) \) are coprime irreducible polynomials (see Claim 6.1). Claim 6.2 shows that the third property of a pure product is also satisfied with probability \( 1 - (wn)^{-\Omega(1)} \). We analyze Algorithm 10 assuming that \( F \) is a pure product over \( \mathbb{F} \) (which also implies that \( F \) is a pure product over \( \mathbb{L} \)). The third property of a pure product will be used only in Observation 6.5 in Section 6.3.2. The algorithm has three main stages:

1. **Computing the irreducible factors of \( \det(F) \) (Steps 2–6):** From blackbox access to the entries of \( F \), a blackbox access to \( \det(F) \) is computed in \( (wn \log q)^{O(1)} \) time using

\(^1\)with a mild condition on the characteristic and the size of the finite field
Algorithm 10 Average-case matrix factorization

INPUT: Blackbox access to $w^2$, $n$ variate, degree $d$ polynomials $\{f_{st}\}_{s,t\in[w]}$ that constitute the entries of a random $(w, d, n)$-matrix product $F = X_1 \cdot X_2 \cdots X_d$.

OUTPUT: Linear matrices $Y_1, Y_2, \ldots, Y_d$ over $\mathbb{L}$ such that $F = Y_1 \cdot Y_2 \cdots Y_d$.

1: /* Factorization of the determinant */
2: Compute blackbox access to $\det(F)$.
3: Compute blackbox access to the irreducible factors of $\det(F)$; call them $g_1, g_2, \ldots, g_d$.
4: if the number of irreducible factors is not equal to $d$ then
5: Output 'Failed'.
6: end if
7:
8: /* Affine equivalence test for determinant */
9: Set $j = 1$.
10: while $j \leq d$ do
11: Call the algorithm in Theorem 6.1 with input as blackbox access to $g_j$; let $B_j$ and $b_j$ be its output. Construct the $w \times w$ full-rank linear matrix $Z_j$ over $\mathbb{L}$ determined by $B_j$ and $b_j$.
12: if the algorithm outputs ‘$g_j$ not affine equivalent to $\Det_w$’ then
13: Output 'Failed'.
14: end if
15: Set $j = j + 1$.
16: end while
17:
18: /* Rearrangement of the matrices */
19: Call Algorithm 13 on input blackbox access to $F$ and $Z_1, \ldots, Z_d$, and let $Y_1, \ldots, Y_d$ be its output.
20: if Algorithm 13 outputs ‘Rearrangement not possible’ then
21: Output 'Failed'.
22: end if
23:
24: Output $Y_1, Y_2, \ldots, Y_d$. 
2. **Affine equivalence test (Steps 9–16):** Let \( j = \sigma(i) \) and \( X'_i \) be the matrix \( X_i \) with the affine forms in the first row multiplied by \( c_i \). Then, \( g_j = \det(X'_i) = c_i \cdot \det(X_i) \), which is affine equivalent to \( \det_w \). At step 11, the algorithm in Theorem 6.1 (given in Section 6.5) finds a \( B_j \in \mathbb{L}^{w^2 \times n} \) of rank \( w^2 \) and \( b_j \in \mathbb{L}^{w^2} \) such that \( g_j = \det_w(B_j x + b_j) \), with probability \( 1 - (wdn)^{-\Omega(1)} \). Let \( Z_j \) be the matrix obtained by appropriately replacing the entries of the \( w \times w \) symbolic matrix with the affine forms in \( B_j x + b_j \) such that \( \det(Z_j) = g_j = \det(X'_i) \). This certifies that there are matrices \( C_i, D_i \in SL(w, \mathbb{L}) \) satisfying, \( Z_j = C_i \cdot X'_i \cdot D_i \) or \( Z_j^T = C_i \cdot X'_i \cdot D_i \) (see Theorem 2.1 in Section 2.6). Multiplying the first column of \( C_i \) with \( c_i \) and calling the resulting matrix \( C_i \) again, we see that there are matrices \( C_i, D_i \in GL(w, \mathbb{L}) \) satisfying, \( Z_j = C_i \cdot X_i \cdot D_i \) or \( Z_j^T = C_i \cdot X_i \cdot D_i \). Observe that such \( C_i, D_i \) are unique up to multiplications by elements in \( \mathbb{L}^\times \) i.e., if \( C_i \cdot X_i \cdot D_i = C'_i \cdot X_i \cdot D'_i \), where \( X_i \) is a full-rank matrix, then \( C'_i = \alpha C_i \) and \( D'_i = \alpha^{-1} D_i \) for some \( \alpha \in \mathbb{L}^\times \).

3. **Rearrangement of the retrieved matrices (Steps 19–22):** This stage is the most crucial part of Algorithm 10. At step 19, Algorithm 13 constructs the matrices \( Y_1, Y_2, \ldots, Y_d \) by determining the permutation \( \sigma \) and whether \( Z_{\sigma(i)} = C_i \cdot X_i \cdot D_i \) or \( Z_{\sigma(i)}^T = C_i \cdot X_i \cdot D_i \). Internally, Algorithm 13 uses Algorithm 14, which when given blackbox access to \( F_d = F \) and a \( Z \) (that is either \( Z_k \) or \( Z_k^T \) for some \( k \in [d] \)), does the following with probability \( 1 - (wdn)^{-\Omega(1)} \): If \( Z = C_d \cdot X_d \cdot D_d \) then it outputs a \( \hat{D}_d = a_d D_d \) for some \( a_d \in \mathbb{L}^\times \). For all other cases – if \( Z = C_i \cdot X_i \cdot D_i \) or \( Z^T = C_i \cdot X_i \cdot D_i \) for \( i \in [d - 1] \), or \( Z^T = C_d \cdot X_d \cdot D_d \) – it outputs 'Failed'. Algorithm 14 uses the critical fact that \( F \) is a pure product to accomplish the above and locate the unique last matrix. The running time of the algorithm, which is \((wdn \log q)^{O(1)}\), and its proof of correctness (which also gives the uniqueness of factorization mentioned in the remark after Theorem 1.2) are discussed in Section 6.3.2. Algorithm 13 calls Algorithm 14 on inputs \( F, Z_k \) and \( F, Z_k^T \) for all \( k \in [d] \). If Algorithm 14 returns a matrix \( \hat{D}_d \) for some \( k \in [d] \) on either inputs \( F, Z_k \) or \( F, Z_k^T \) then it sets \( M_d = Z_k \) or \( M_d = Z_k^T \) respectively, and \( \sigma(d) = k \). Subsequently, Algorithm 13 computes blackbox access to a length \( d - 1 \) matrix product \( F_{d-1} = F \cdot \hat{D}_d \cdot M_d^{-1} = X_1 \cdots X_{d-2} \cdot (X_{d-1} \cdot a_d C_d^{-1}) \), and repeats the above process.
to compute $M_{d-1}$ and $\sigma(d-1)$ with the inputs $F_{d-1}$ and \{Z_1, \ldots, Z_d\}\{Z_{\sigma(d)}\). Thus, using Algorithm 14 repeatedly, Algorithm 13 iteratively determines $\sigma$ and $M_d, M_{d-1}, \ldots, M_2$.

At the $(d-t+1)$-th iteration, for $t \in [d-1, 2]$, it computes a matrix $\tilde{D}_t = a_t(C_{t+1} \cdot \tilde{D}_t)$ for some $a_t \in \mathbb{L}^\times$, sets $M_t$ and $\sigma(t)$ accordingly, creates blackbox access to $F_{t-1} = F_1 \cdot \tilde{D}_1 \cdot M_1^{-1}$ and prepares the list \{Z_1, \ldots, Z_d\}\{Z_{\sigma(d)}, Z_{\sigma(d-1)}, \ldots, Z_{\sigma(t)}\} for the next iteration. Finally, setting $Y_1 = F_1$ and $Y_i = M_i \cdot \tilde{D}_i^{-1}$, for all $i \in [2, d]$, we have $F = \prod_{i=1}^d Y_i$.

6.1.1.2 Analysis of Algorithm 11

Let $f$ be the polynomial computed by a $(w, d, n)$-ABP $X_1 \cdot X_2 \cdots X_d$. We can assume that $f$ is a homogeneous degree-$d$ polynomial and the entries in each $X_i$ are linear forms (i.e., affine forms with constant term zero), owing to the following simple homogenization trick.

**Homogenization of ABP:** Consider the $(n+1)$-variate homogeneous degree-$d$ polynomial

$$f_{\text{hom}} = x_0^d \cdot f \left( \frac{x_1}{x_0}, \frac{x_2}{x_0}, \ldots, \frac{x_n}{x_0} \right).$$

The polynomial $f_{\text{hom}}$ is computable by the $(w, d, n)$-ABP $\tilde{X}_1 \cdot \tilde{X}_2 \cdots \tilde{X}_d$, where $\tilde{X}_i$ is equal to $X_i$ but with the constant term in the affine forms multiplied by $x_0$. If we construct an ABP for $f_{\text{hom}}$ then an ABP for $f$ is obtained by setting $x_0 = 1$. We give an overview of the three main stages in Algorithm 11. As in Algorithm 10, the matrices $X_1, X_2, \ldots, X_d$ are assumed to be full-rank linear matrices and further, for a similar reason, the $2w$ linear forms in $X_1$ and $X_d$ are assumed to be $\mathbb{F}$-linearly independent. For a field $\mathbb{K} \supseteq \mathbb{F}$, we say $f$ is zero modulo a $\mathbb{K}$-linear space $\mathcal{X} = \text{span}_{\mathbb{K}}\{l_1, \ldots, l_w\}$, where the $l_i$ are linear forms in $\mathbb{K}[x]$, if $f$ is in the ideal of $\mathbb{K}[x]$ generated by $\{l_1, \ldots, l_w\}$. This is also denoted by $f = 0 \mod \langle l_1, \ldots, l_w \rangle$.

1. **Computing the corner spaces (Steps 2–6):** Polynomial $f$ is zero modulo each of the two $w$-dimensional $\mathbb{F}$-linear spaces $\mathcal{X}_1$ and $\mathcal{X}_d$ spanned by the linear forms in $X_1$ and $X_d$ respectively. We show in Lemma 6.2, if $n \geq 4w^2$ then with probability $1 - (wdn)^{-\Omega(1)}$ the following holds: Let $\mathbb{K} \supseteq \mathbb{F}$ be any field. If $f = 0 \mod \langle l_1, \ldots, l_w \rangle$, where the $l_i$ are linear forms in $\mathbb{K}[x]$, then the $l_i$ either belong to the $\mathbb{K}$-span of the linear forms in $X_1$ or belong to the $\mathbb{K}$-span of the linear forms in $X_d$. In this sense, the spaces $\mathcal{X}_1$ and $\mathcal{X}_d$ are unique. The algorithm invokes Algorithm 15 which computes bases of $\mathcal{X}_1$ and $\mathcal{X}_d$ by solving $O(n)$ systems of polynomial equations over $\mathbb{F}$. Such a system has $d^{O(w^2)}$ equations in $m = O(w^3)$ variables and the degree of the polynomials in the system is at most $d$; we intend to find all the solutions in $\overline{\mathbb{F}}^m$. It turns out that owing to the uniqueness of $\mathcal{X}_1$ and $\mathcal{X}_d$, the variety over $\overline{\mathbb{F}}$ (the algebraic closure of $\mathbb{F}$) defined by such
Algorithm 11 Average-case ABP reconstruction

INPUT: Blackbox access to a $n$ variate, degree $d$ polynomial $f$ computed by a random $(w, d, n)$-ABP.
OUTPUT: A $(w, d, n)$-ABP over $\mathbb{L}$ computing $f$.

1: /* Computing the corner spaces */
2: Call Algorithm 15 on $f$ to compute bases of the two unique $w$-dimensional $\mathbb{F}$-linear spaces $X_1$ and $X_d$, spanned by linear forms in $\mathbb{F}[x]$, such that $f$ is zero modulo each of $X_1$ and $X_d$.
3: if Algorithm 15 outputs ‘Failed’ then
4: Output ‘Failed to construct an ABP’.
5: end if
6: Compute a transformation $A \in \text{GL}(n, \mathbb{F})$ that maps the bases of $X_1$ and $X_d$ to distinct variables $y = \{y_1, y_2, \ldots, y_w\}$ and $z = \{z_1, z_2, \ldots, z_w\}$ respectively, where $y, z \subseteq x$. Let $r = x \setminus (y \cup z)$, $X'_1 = (y_1 y_2 \ldots y_w)$, $X'_d = (z_1 z_2 \ldots z_w)^T$ and $f' = f(Ax)$.
7: /* Computing the coefficients of the $r$ variables in the intermediate matrices */
8: Construct blackbox access to the $w^2$ polynomials that constitute the entries of the $w \times w$ matrix $F = (\frac{\partial f'}{\partial y_s z_t} |_{y=0,z=0})_{s,t \in [w]}$.
9: Call Algorithm 10 on input $F$ to compute a factorization of $F$ as $S_2 \cdot S_3 \cdots S_{d-1}$.
10: if Algorithm 10 outputs ‘Failed’ then
11: Output ‘Failed to construct an ABP’.
12: end if
13: /* Computing the coefficients of the $y$ and $z$ variables in the intermediate matrices */
14: Call Algorithm 16 on inputs $f'$ and $\{S_2, S_3, \ldots, S_{d-1}\}$ to compute matrices $T_2, T_3, \ldots, T_{d-1}$ such that $f'$ is computed by the ABP $X'_1 \cdot T_2 \cdots T_{d-1} \cdot X'_d$.
15: if Algorithm 16 outputs ‘Failed’ then
16: Output ‘Failed to construct an ABP’.
17: end if
18: Apply the transformation $A^{-1}$ on the $x$ variables in the matrices $X'_1, X'_d$, and $T_k$ for $k \in [2, d - 1]$. Call the resulting matrices $Y_1, Y_d$, and $Y_k$ for $k \in [2, d - 1]$ respectively.
19: Output $Y_1 \cdot Y_2 \ldots Y_d$ as the ABP computing $f$. 

113
a system has exactly two points and these points lie in \( \mathbb{F}^m \). From the two solutions, bases of \( X_1 \) and of \( X_d \) can be derived. The two solutions of the system are computed by a randomized algorithm running in \((dw^3 \log q)^{O(1)}\) time ([Ier89, HW99], see Lemma 2.5) – the algorithm exploits the fact that the variety over \( \overline{\mathbb{F}} \) is zero-dimensional. Thus, at step 2, the two spaces are either equal to \( X_1 \) and \( X_d \) or \( X_1 \) and \( X_d \) respectively. Without loss of generality, we assume the former. Once bases of the corner spaces \( X_1 \) and \( X_d \) are computed, an invertible transformation \( A \) maps the linear forms in the bases to distinct variables (as the linear forms in \( X_1 \) and \( X_d \) are \( \mathbb{F} \)-linearly independent).

2. **Computing the coefficients of the \( r \) variables (Steps 9–13):** There is an ABP \( X'_1 \cdot X'_2 \ldots X'_d \) computing \( f' = f(Ax) \), where \( X'_i \) and \( X'_d \) are equal to \( (y_1 y_2 \ldots y_w) \) and \( (z_1 z_2 \ldots z_w)^T \) respectively. For \( k \in [2, d - 1] \), let \( R_k = (X'_k)_{y=0,x=0} \) and \( F = R_2 \cdot R_3 \ldots R_{d-1} \). As \( X_1 \cdot X_2 \cdots X_d \) is a random \((w,d,n)\)-ABP, \( R_2 \cdot R_3 \ldots R_{d-1} \) is a random \((w,d-2,n-2w)\)-matrix product over \( \mathbb{F} \). The \((s,t)\)-th entry of \( F \) is equal to \( \left( \frac{\partial f'}{\partial y_s z_t} \right)_{y=0,z=0} \), for \( s, t \in [w] \).

Blackbox access to each of the \( w^2 \) entries of \( F \) are constructed in \((wdn \log q)^{O(1)}\) time using Claim 2.2. From \( F \), Algorithm 10 computes linear matrices \( S_2, \ldots, S_{d-1} \) over \( \mathbb{L} \) in \( r = x \setminus (y \cup z) \) variables such that \( F = S_2 \cdot S_3 \ldots S_{d-1} \). Moreover, the uniqueness of factorization implies there are linear matrices \( T_2, \ldots, T_{d-1} \) over \( \mathbb{L} \) in the \( x \)-variables, satisfying \( (T_k)_{y=0,z=0} = S_k \), such that \( f' \) is computed by the ABP \( X'_1 \cdot T_2 \cdots T_{d-1} \cdot X'_d \).

3. **Computing the coefficients of \( y \) and \( z \) variables in \( T_k \) (Steps 16–20):** Algorithm 16 finds the coefficients of the \( y \) and \( z \) variables in the linear forms present in \( T_2, \ldots, T_{d-1} \) in \((wdn \log q)^{O(1)}\) time. We present the idea here; the detail proof of correctness is given in Section 6.4.2. In the following discussion, \( M(i,j) \) denotes the \((i,j)\)-th entry, \( M(i,* \) \) the \( i \)-th row, and \( M(*,j) \) the \( j \)-th column of a linear matrix \( M \). Let us focus on finding the coefficients of \( y_1 \) in the linear forms present in \( T_2(1,*), T_3, \ldots, T_{d-2}, T_{d-1}(*,1) \). There are \( w^2(d-4) + 2w \) linear forms in these matrices and these would be indexed by \([w^2(d-4) + 2w]\). Let \( c_e \) be the coefficient of \( y_1 \) in the \( e \)-th linear form \( f_e \) for \( e \in [w^2(d-4) + 2w] \).

We associate a polynomial \( h_e(r) \) in \( r \) variables with \( l_e \) as follows: If \( l_e \) is the \((i,j)\)-th entry of \( T_k \) then \( h_e = [S_2(1,* \) \) \( \cdot S_3 \cdots S_{k-2} \cdot S_{k-1}(*,i)] \cdot [S_{k+1}(j,* \) \) \( \cdot S_{k+2} \cdots S_{d-2} \cdot S_{d-1}(*,1)] \), by identifying the \( 1 \times 1 \) matrix of the R.H.S with the entry of the matrix. Observe that if \( f' \) is treated as a polynomial in \( y \) and \( z \) variables with coefficients in \( \mathbb{L}(r) \) then the coefficient of \( y_1^2 z_1 \) is exactly \( \sum_{e \in [w^2(d-4) + 2w]} c_e \cdot h_e(r) \). On the other hand, this coefficient is \( \left( \frac{\partial f'}{\partial y_1^2 z_1} \right)_{y=0,z=0} \), for which we can obtain blackbox access using Claim 2.2. This allows
us to write the equation,
\[
\sum_{e=1}^{w^2(d-4)+2w} c_e \cdot h_e(r) = \left( \frac{\partial f'}{\partial y_1^2 z_1} \right)_{y=0, z=0}.
\] (6.1)

For \( d \geq 5 \), we show in Lemma 6.3 and Corollary 6.1 that the polynomials \( h_e \), for \( e \in [w^2(d-4)+2w] \), are \( \mathbb{L} \)-linearly independent with probability \( 1 - (w^2 d n)^{-\Omega(1)} \), over the randomness of the input \( f \). By substituting random values to the \( r \) variables in the above equation, we can set up a system of \( w^2(d-4)+2w \) independent linear equations in the \( c_e \) variables. The linear independence of the \( h_e \) polynomials ensures that we can solve for \( c_e \) (by Claim 2.3).

**6.1.1.3 Analysis of Algorithm 12**

Algorithm 12 Determinant equivalence test

**INPUT:** Blackbox access to an \( n \) variate, degree \( w \) polynomial \( f \), where \( n \geq w^2 \).
**OUTPUT:** If \( f \) is affine equivalent to \( \text{Det}_w \) then output a \( B \in \mathbb{L}^{w^2 \times n} \) of rank \( w^2 \) and a \( b \in \mathbb{L}^{w^2} \) such that \( f = \text{Det}_w(Bx + b) \), else output ‘\( f \) not affine equivalent to \( \text{Det}_w \)’.

1: /* Reduction to equivalence testing */
2: Use ‘variable reduction’ (Algorithm 1) and ‘translation equivalence’ (Algorithm 2) to compute a \( (w^2, w) \)-polynomial that is equivalent to \( \text{Det}_w \) if and only if \( f(x) \) is affine projection of \( \text{Det}_w \) via a full-rank transformation.
3: /* The above step succeeds with high probability. Reusing symbols, the input to the next step is blackbox access to a \( (w^2, w) \)-polynomial \( f(x) \). */
4: /* Reduction to Permutation-Scaling (PS)-Equivalence */
6: Use Algorithm 17 to compute a \( D \in \text{GL}(w^2, \mathbb{L}) \) such that \( f(Dx) \) is PS-equivalent to \( \text{Det}_w \) if and only if \( f(x) \) is equivalent to \( \text{Det}_w \). /* This step succeeds with high probability. */
7: 8: /* Doing the PS-equivalence */
9: This step follows from [Kay12a].

Algorithm 12 has three stages:

1. **Reduction to equivalence testing:** Applying known techniques – ‘variable reduction’ (Algorithm 1) and ‘translation equivalence’ (Algorithm 2) – the affine projection via full-rank testing problem is reduced in randomized polynomial time to equivalence testing for \( \text{Det}_w \) with high probability. A \( (w^2, w) \)-polynomial \( g(x) \) is equivalent to \( \text{Det}_w \) if there exists a \( Q \in \text{GL}(w^2, \mathbb{F}) \) such that \( g = \text{Det}_w(Qx) \). An equivalence test takes blackbox access to a \( (w^2, w) \)-polynomial \( g(x) \) as input and does the following with high probability:
If \( g \) is equivalent to \( \text{Det}_w \) then it outputs a \( Q \in \text{GL}(w^2, \mathbb{L}) \) such that \( g = \text{Det}_w(Qx) \) else it outputs ‘\( g \) not equivalent to \( \text{Det}_w \)’. From such a \( Q \), a required \( B \in \mathbb{L}^{w^2 \times n} \) of rank \( w^2 \) and a \( b \in \mathbb{L}^{w^2} \) satisfying \( f = \text{Det}_w(Bx + b) \) can be computed easily. For the next two stages we set \( n = w^2 \), and reusing symbols the input to the next stage is blackbox access to a \((w^2, w)\)-polynomial \( f(x) \). The goal in the next two stages is to determine whether \( f(x) \) is equivalent to \( \text{Det}_w \).

2. Reduction to Permutation-Scaling (PS)-Equivalence: At this stage the equivalence testing problem is reduced to a simpler problem of testing whether a polynomial given as blackbox is permutation-scaling (PS)-equivalent to \( \text{Det}_w \). A \((w^2, w)\) polynomial \( g(x) \) is PS-equivalent to \( \text{Det}_w \) if there is a permutation matrix \( P \) and a diagonal matrix \( S \) such that \( g = \text{Det}_w(PSx) \). The reduction is given in Algorithm 17. The algorithm proceeds by computing an \( \mathbb{F} \)-basis of the Lie algebra of the group of symmetries of \( f \) (denoted as \( \mathfrak{g}_f \), see Lemma 2.4). It then picks an element \( F \) uniformly at random from \( \mathfrak{g}_f \) and computes its characteristic polynomial \( h(x) \). Since \( F \in \mathfrak{g}_f \), it is similar to a \( L \in \mathfrak{g}_{\text{Det}_w} \) (see Claim 2.1 in Section 6.5.1), implying that their characteristic polynomials are equal. As \( F \) is a random element of \( \mathfrak{g}_f \), \( L \) is also a random element of \( \mathfrak{g}_{\text{Det}_w} \). In Lemma 6.5, we show that the characteristic polynomial \( h \) of a \( L \in \mathfrak{g}_{\text{Det}_w} \) is square-free and splits completely over \( \mathbb{L} \), with high probability. (This lemma makes our reduction to PS-equivalence simpler than [Kay12a], enabling the equivalence test to work over finite fields.) The roots of \( h \) are computed in randomized \( (w \log q)^{O(1)} \) time ([CZ81], see also [vzGG03]). From the roots, a \( D \in \text{GL}(w^2, \mathbb{L}) \) can be computed such that \( D^{-1}FD \) is diagonal. Thereafter, the structure of the group of symmetries of \( \text{Det}_w \) and its Lie algebra helps argue, in Section 6.5.2, that \( f(Dx) \) is PS-equivalent to \( \text{Det}_w \).

3. Doing the PS-equivalence: This step follows directly from [Kay12a] (see Lemma 6.4).

6.1.2 Dependencies between the algorithms

Figure 6.1 illustrates the dependencies between the three main algorithms. Algorithm 11 calls Algorithm 10 on input \( F \), which is a random \((w, d - 2, n)\)-matrix product. With high probability Algorithm 10 returns a factorization of \( F \) as \( S_2 \cdot S_3 \cdots S_{d-1} \) to Algorithm 11. Similarly Algorithm 10 calls Algorithm 12 \( d \) times on inputs \( g_1, g_2, \ldots, g_d \) respectively, where the \( g_j \) are \((n, w)\) polynomials. For every \( j \in [d] \), with high probability Algorithm 12 returns a \( B_j \in \mathbb{L}^{w^2 \times n} \) of rank \( w^2 \) and a \( b_j \in \mathbb{L}^{w^2} \) satisfying \( g_j = \text{Det}_w(B_jx + b_j) \) to Algorithm 10.

\[1\] In [Kay12a], a basis of the centralizer of \( F \) in \( \mathfrak{g}_f \) is computed first and then a \( D \in \text{GL}(w^2, \mathbb{C}) \) is obtained that simultaneously diagonalizes this basis.
6.2 Purity of random matrix product

In this section we prove that a random \((w, d, n)\)-matrix product is a pure product with high probability if \(n \geq w^2\). Claim 6.1 shows that a random matrix product satisfies the first two conditions of a pure product with high probability.

**Claim 6.1** Let \(X_1 \cdot X_2 \cdots X_d\) be a random \((w, d, n)\)-matrix product over \(\mathbb{F}\). If \(n \geq w^2\) then \(X_1, X_2, \ldots, X_d\) are full-rank linear matrices and \(\det(X_1), \det(X_2), \ldots, \det(X_d)\) are coprime irreducible polynomials with probability \(1 - \frac{(w^2)^O(1)}{q}\).

**Proof:** Let \(Y = (l_{st})_{s,t \in [w]}\) be a linear matrix, where \(l_{st} = \sum_{i=1}^{n} c_{st,i} x_i\). Treat the \(w^2n\) coefficients, \(c = \{c_{st,i} \mid s, t \in [w], i \in [n]\}\) as formal variables. The matrix \(X_i\), for all \(i \in [d]\) is constructed by choosing the values of \(c\) variables independently and uniformly at random from \(\mathbb{F}\). Consider the \(w^2 \times n\) coefficient matrix of a \(w \times w\) linear matrix where the rows are indexed by the linear forms, columns are indexed by the variables and the \((s, t)\)-th entry is the coefficient of the variables indexing the \(t\)-th column in linear form indexing the \(s\)-th row.

It is trivial to see that the rank of the coefficient matrix is equal to the dimension of the linear space spanned by the linear forms in the linear matrix. For \(n \geq w^2\), the rank of the coefficient matrix of \(M\) is \(w^2\) over \(\mathbb{F}(c)\). Hence, by union bound and Claim 2.3 with probability at least \(1 - \frac{d^2w}{q}\), rank of the coefficient matrix of \(X_i\), for all \(i \in [d]\) is \(w^2\). Since \(\text{Det}_w\) is an irreducible polynomial, with probability at least \(1 - \frac{d^2w}{q}\), \(\det(X_1), \det(X_2), \ldots, \det(X_d)\) are distinct irreducible polynomials. ■

The following claim implies that a random matrix product satisfies the third property of a pure product with high probability.

**Claim 6.2** If \(E = Q_1 \cdots Q_\ell\) is a random \((w, \ell, m)\)-matrix product over \(\mathbb{F}\), where \(w^2 + 1 \leq m \leq n\) and \(\ell \leq d\), then the entries of \(E\) are \(\mathbb{F}\)-linearly independent with probability \(1 - \frac{(w^2)^O(1)}{q}\).
Proof: Treat the coefficients of the linear forms in \( Q_1, Q_2, \ldots, Q_\ell \) as distinct formal variables. In particular
\[
Q_k = \sum_{i=1}^{m} U_i^{(k)} x_i \quad \text{for} \quad k \in [\ell],
\]
where the \( U_i^{(k)} \) are \( w \times w \) matrices and the entries of these matrices are distinct \( u \)-variables. The entries of the matrix product \( E \) are polynomials in the \( x \)-variables over \( \mathbb{F}(u) \). If we show the \( w^2 \) entries of \( E \) are \( \mathbb{F}(u) \)-linearly independent then an application of Schwartz-Zippel lemma implies the statement of the claim. On the other hand, to show that the entries of \( E \) are \( \mathbb{F}(u) \)-linearly independent, it is sufficient to show that the entries are \( \mathbb{F} \)-linearly independent under a setting of the \( u \)-variables to \( \mathbb{F} \) elements. Consider such a setting: For every \( k \in [\ell] \setminus \{1\} \), let \( U_{k+1}^{(k)} = I_w \) and \( U_i^{(k)} = 0 \) for all \( i \in [m] \setminus \{w^2 + 1\} \). Let \( U_1^{(1)} = 0 \) for all \( i \geq w^2 + 1 \) and set \( U_1^{(1)}, \ldots, U_{w^2}^{(1)} \) in a way so that the linear forms in \( \sum_{w^2+1}^{w^2} U_i^{(1)} x_i \) are \( \mathbb{F} \)-linearly independent. It is straightforward to check that the entries of \( E \) under this setting are \( \mathbb{F} \)-linearly independent.

6.3 Average-case linear matrix factorization: Proof of Theorem 1.2

The algorithm in Theorem 1.2 is presented in Algorithm 10. To complete the analysis, given in Section 6.1.1.1, we need to argue the correctness of the key step of rearrangement of the matrices (Algorithm 13) by finding the last matrix (Algorithm 14). As the functioning of Algorithm 13 is already sketched out in Section 6.1.1.1, the reader may skip to Section 6.3.2. For completeness, we include an analysis of Algorithm 13 in the following subsection.

6.3.1 Rearranging the matrices

Recall, we have assumed \( F \) is a pure \((w, d, n)\)-matrix product \( X_1 \cdot X_2 \cdots X_d \) over \( \mathbb{F} \), and hence also a pure product over \( \mathbb{L} \). The inputs to Algorithm 13 are \( d \) full-rank linear matrices \( Z_1, Z_2, \ldots, Z_d \) over \( \mathbb{L} \) such that there are matrices \( C_i, D_i \in \text{GL}(w, \mathbb{L}) \) and a permutation \( \sigma \) of \([d]\) satisfying \( Z_{\sigma(i)} = C_i \cdot X_i \cdot D_i \) or \( Z_{\sigma(i)}^T = C_i \cdot X_i \cdot D_i \) for every \( i \in [d] \). Algorithm 13 iteratively determines \( \sigma \) (implicitly) by repeatedly using Algorithm 14. The behavior of Algorithm 14 is summarized in the lemma below. For the lemma statement, assume \( n \geq 2w^2 \), \( Z \) is a full-rank linear matrix over \( \mathbb{L} \), and \( F_i \) is a pure \((w, t, n)\)-matrix product \( R_1 \cdot R_2 \cdots R_t \) over \( \mathbb{L} \), where \( t \leq d \). Further, there are matrices \( C, D \in \text{GL}(w, \mathbb{L}) \) and \( i \in [t] \) such that \( Z = C \cdot R_i \cdot D \) or \( Z^T = C \cdot R_i \cdot D \).

Lemma 6.1 Algorithm 14 takes input \( Z \) and blackbox access to the \( w^2 \) entries of \( F_i \), and with
Algorithm 13 Rearrangement of the matrices

INPUT: Blackbox access to $F$, and $w \times w$ full-rank linear matrices $Z_1, Z_2, \ldots, Z_d$ over $\mathbb{L}$.
OUTPUT: Linear matrices $Y_1, Y_2, \ldots, Y_d$ over $\mathbb{L}$ such that $F = Y_1 \cdot Y_2 \cdots Y_d$.

1: Set $t = d, k = 1$, and $F_d = F$.
2: while $t > 1$ do
3: 
4: while $k \leq t$ do
5: 
6: Call Algorithm 14 on inputs $F_t$ and $Z_k$.
7: 
8: if Algorithm 14 outputs $\tilde{D}$ then
9: 
10: /* Such a $\tilde{D} = a_t D_t$ for some $a_t \in \mathbb{L}^\times$. */
11: Rename $Z_k$ as $Z_t$ and $Z_t$ as $Z_k$, and set $\tilde{D}_t = \tilde{D}$. /* $\sigma$ is determined implicitly. */
12: Set $M_t = Z_t$ and $F_{t-1} = F_t \cdot \tilde{D}_t \cdot M_t^{-1}$.
13: Set $k = 1$ and $t = t - 1$.
14: Exit the (inner) loop.
15: end if
16: 
17: Call Algorithm 14 on inputs $F_t$ and $Z_k^T$.
18: if Algorithm 14 outputs a $\tilde{D}$ then
19: 
20: /* Such a $\tilde{D} = a_t D_t$ for some $a_t \in \mathbb{L}^\times$. */
21: Rename $Z_k$ as $Z_t$ and $Z_t$ as $Z_k$, and set $\tilde{D}_t = \tilde{D}$. /* $\sigma$ is determined implicitly. */
22: Set $M_t = Z_t^T$ and $F_{t-1} = F_t \cdot \tilde{D}_t \cdot M_t^{-1}$.
23: Set $k = 1$ and $t = t - 1$.
24: Exit the (inner) loop.
25: end if
26: 
27: end while
28: 
29: if $k = t + 1$ then
30: Exit the (outer) loop.
31: end if
32: 
33: end while
34: 
35: if $t \geq 2$ then
36: Output 'Rearrangement not possible'.
37: else
38: Set $Y_1 = F_1$, and $Y_t = M_t \cdot \tilde{D}_t^{-1}$ for all $t \in [2, d]$. Output $Y_1, \ldots, Y_d$.
39: end if
probability \(1 - \frac{(\log d)^{O(1)}}{q}\) does this: If \(Z = C \cdot R_t \cdot D\) then it outputs a \(\tilde{D} = aD\) for an \(a \in \mathbb{L}^\times\), and for all other cases \(Z = C \cdot R_t \cdot D\) or \(Z^T = C \cdot R_t \cdot D\) for \(i \in [t-1]\), or \(Z^T = C \cdot R_t \cdot D\) – it outputs 'Failed'.

Algorithm 14 and the proof of Lemma 6.1 are presented in Section 6.3.2. We analyze Algorithm 13 below by tracing its steps:

**Step 2:** The algorithm enters an outer loop and iterates from \(t = d\) to \(t = 2\). For a fixed \(t \in [d, 2]\) at the start of the loop the following conditions are satisfied: a) \(F_i\) is a pure \((w, t, n)\)-matrix product \(R_1 \cdot R_2 \cdots R_t\) over \(\mathbb{L}\), here for \(t = d, R_i = X_i\) for all \(i \in [d]\). b) There is a permutation \(\sigma_t\) of \([t]\), and for every \(i \in [t]\) there are matrices \(C_i, D_i \in \text{GL}(w, \mathbb{L})\) such that either \(Z_{\sigma_t(i)} = C_i \cdot R_i \cdot D_i\) or \(Z_{\sigma_t(i)}^T = C_i \cdot R_i \cdot D_i\). In the loop, the algorithm determines \(\sigma_t(t)\) and whether \(Z_{\sigma_t(t)} = C_t \cdot R_t \cdot D_t\) or \(Z_{\sigma_t(t)}^T = C_t \cdot R_t \cdot D_t\).

**Steps 4–23:** Inside the inner loop, the algorithm calls Algorithm 14 on inputs \(F_t, Z_k\) (step 5) and \(F_t, Z^T_k\) (step 13) for all \(k \in [t]\). By Lemma 6.1, only when \(k = \sigma_t(t)\), Algorithm 14 returns a \(\tilde{D} = a_t D_t\) for some \(a_t \in \mathbb{L}^\times\). The renaming of \(Z_k\) and \(Z_t\) (in steps 7 and 15) ensures that we have a suitable permutation \(\sigma_{t-1}\) of \([t-1]\) in the next iteration of the outer loop. The setting of \(M_t\) (in steps 8 and 16) implies that \(M_t = C_t \cdot R_t \cdot D_t\). Hence,

\[ F_{t-1} = F_t \cdot \tilde{D}_t \cdot M_t^{-1} = (R_1 \cdot R_2 \cdots R_{t-1}) \cdot (a_t C_t^{-1}). \]

Note that \(F_{t-1}\) is a pure \((w, t-1, n)\)-matrix product over \(\mathbb{L}\). By reusing symbols and calling \(R_{t-1} \cdot (a_t C_t^{-1})\) as \(R_{t-1}^{-1}\), and \(a_t^{-1} C_t \cdot D_{t-1}\) as \(D_{t-1}\), we observe that the two conditions at step 2 are satisfied in the next iteration of the outer loop.

**Step 34:** As \(F_{t-1} = F_t \cdot \tilde{D}_t \cdot M_t^{-1}\) at every iteration of the outer loop, setting \(Y_t = M_t \cdot \tilde{D}_t^{-1}\) implies \(F_{t-1} = F_t \cdot Y_t^{-1}\) for every \(t \in [d, 2]\). Therefore, \(F = F_d = Y_1 \cdots Y_d\).

### 6.3.2 Determining the last matrix: Proof of Lemma 6.1

We give an overview of the proof by first assuming that \(Z\) is the ‘last’ matrix in the pure product \(F_t\). The correctness of the idea is then made precise by tracing the steps of Algorithm 14.

**Overview:** Suppose \(Z = C \cdot R_t \cdot D\), where \(C, D \in \text{GL}(w, \mathbb{L})\). As \(Z\) is a full-rank linear matrix, we can assume the entries of \(Z\) are distinct variables, by applying an invertible linear transformation. For any polynomial \(h \in \mathbb{L}[x], \ h \mod \det(Z)\) can be identified with an element of \(\mathbb{L}(x)\).
This is because, $\det(Z)$ is multilinear and so there is an injective ring homomorphism from $\mathbb{L}[x]/(\det(Z))$ to $\mathbb{L}(x)$ via a simple substitution map taking a variable to a rational function. Let $Z', F'_t \in \mathbb{L}(x)^{w \times w}$ be obtained by reducing the entries of $Z$ and $F_t$, respectively, modulo $\det(Z)$. The coprimality of the determinants of $R_1, \ldots, R_t$ and their full-rank nature imply,

$$D \cdot \text{Kernel}_{\mathbb{L}(x)}(Z') = \text{Kernel}_{\mathbb{L}(x)}(F'_t),$$

and these two kernels have dimensions one. A basis of $\text{Kernel}_{\mathbb{L}(x)}(Z')$ can be easily derived as $Z$ is known explicitly. However, we only have blackbox access to $F'_t$. To leverage the above relation, we compute bases of $\text{Kernel}_{\mathbb{L}}(F'_t(a))$ and $\text{Kernel}_{\mathbb{L}}(Z'(a))$ for several random $a \in_r \mathbb{F}^n$, and form two matrices $U, V \in \text{GL}(w, \mathbb{L})$ from these bases so that $D$ equals $U \cdot V^{-1}$ (up to scaling by elements in $\mathbb{L}^\times$). Hereafter, $\text{Kernel}_{\mathbb{L}}$ will also be denoted as $\text{Ker}$ in Algorithm 14 and its analysis.

**Applying an invertible linear map (Step 2):** The invertible linear transformation lets us assume that $Z = (z_{lk})_{l,k \in [w]}$, where the $z_{lk}$ are distinct variables in $x$.

**Reducing $Z$ and $F_t$ modulo $\det(Z)$ (Step 5):** The reduction of the entries of $Z$ and the blackbox entries of $F_t$ modulo $\det(Z)$ is achieved by the substitution,

$$z_{11} = -\frac{\sum_{k=2}^{w} z_{1k} \cdot N_{1k}}{N_{11}},$$

where $N_{lk}$ denotes the $(l, k)$-th cofactor of $Z$ for $l, k \in [w]$. After the substitution, the matrices become $Z'$ and $F'_t = R'_1 \cdot R'_2 \cdots R'_t$ respectively. As there are $i \in [t]$ and $C, D \in \text{GL}(w, \mathbb{L})$ such that either $Z = C \cdot R_i \cdot D$ or $Z^T = C \cdot R_i \cdot D$, we have either $Z' = C \cdot R'_i \cdot D$ or $(Z')^T = C \cdot R'_i \cdot D$ and hence $\det(Z') = \det(R'_i) = \det(F'_i) = 0$.

**Observation 6.1**  
1. $\text{Kernel}_{\mathbb{L}(x)}(Z') = \text{span}_{\mathbb{L}(x)} \{(N_{11} N_{12} \ldots N_{1w})^T\}$,

2. $\text{Kernel}_{\mathbb{L}(x)}((Z')^T) = \text{span}_{\mathbb{L}(x)} \{(N_{11} N_{21} \ldots N_{w1})^T\}$.

Hence, $\text{Kernel}_{\mathbb{L}(x)}(Z')$ has dimension one, and the observation below implies $\text{Kernel}_{\mathbb{L}(x)}(F'_t)$ is also one dimensional. The proof follows from the coprimality of $\det(R_1), \det(R_2), \ldots, \det(R_t)$.

**Observation 6.2** For all $j \in [t]$ and $j \neq i$, $\det(R'_j) \neq 0$, and so the dimension of $\text{Kernel}_{\mathbb{L}(x)}(F'_t)$ is one.
Algorithm 14 Determining the last matrix

**INPUT:** Blackbox access to a \((w, l, n)\)-matrix product \(F_t = R_1 \cdots R_l\) and a full-rank linear matrix \(Z\) over \(\mathbb{L}\).

**OUTPUT:** A matrix \(\hat{D} \in \text{GL}(w, \mathbb{L})\), if \(Z = C \cdot R_l \cdot D\). The output matrix \(\hat{D}\) is equal to \(aD\) for some \(a \in \mathbb{L}^\times\).

1: /* Applying an invertible linear map */
2: Let the first \(w^2\) variables in \(x\) be \(z = \{z_{lk}\}_{l,k \in [w]}\). Compute an invertible linear map \(A\) that maps the affine forms in \(Z\) to distinct \(z\) variables, and apply \(A\) to the \(w^2\) blackbox entries of \(F_t\). Reusing symbols, \(Z = (z_{lk})_{l,k \in [w]}\) and \(F_t\) is the matrix product after the transformation.
3: /* Reducing \(Z\) and \(F_t\) modulo \(\det(Z)\) */
4: Let \(N_{lk}\) be the \((l, k)\)-th cofactor of \(Z\), for \(l, k \in [w]\). Substitute \(z_{11} = \frac{-\sum_{k=2}^w z_{1k} N_{1k}}{N_{11}}\) in \(Z\) and in the blackbox for \(F_t\). Call the matrices \(Z'\) and \(F_t'\) respectively after the substitution.
6: /* Computing the kernels at random points */
8: for \(k = 1\) to \(w + 1\) do
9: Choose \(a_k, b_k \in \mathbb{F}^n\). Compute bases of \(\text{Ker}(F_t'(a_k))\), \(\text{Ker}(Z'(a_k))\), \(\text{Ker}(F_t'(b_k))\), \(\text{Ker}(Z'(b_k))\). Pick non-zero \(u_k \in \text{Ker}(F_t'(a_k))\), \(v_k \in \text{Ker}(Z'(a_k))\), \(w_k \in \text{Ker}(F_t'(b_k))\), \(s_k \in \text{Ker}(Z'(b_k))\). If the computation fails (i.e., \(N_{11}(a_k) = 0\) or \(N_{11}(b_k) = 0\)), or any of the kernels is not one dimensional, output ‘Failed’.
10: end for
12: /* Extracting \(D\) from the kernels */
13: Compute \(\alpha_k, \beta_k, \gamma_k, \delta_k \in \mathbb{L}\) for \(k \in [w]\) such that \(u_{w+1} = \sum_{k=1}^w \alpha_k u_k\), \(v_{w+1} = \sum_{k=1}^w \beta_k v_k\), \(w_{w+1} = \sum_{k=1}^w \gamma_k w_k\) and \(s_{w+1} = \sum_{k=1}^w \delta_k s_k\). If the computation fails, or any of \(\alpha_k, \beta_k, \gamma_k, \delta_k\) is zero for some \(k \in [w]\), output ‘Failed’.
14: Set \(U, V, W, S \in \mathbb{L}^{w \times w}\) such that the \(k\)-th column of \(U, V, W, S\) are \(\frac{\alpha_k u_k}{\beta_k}, v_k, \frac{\gamma_k w_k}{\delta_k}, s_k\) respectively. If any of \(U, V, W, S \not\in \text{GL}(w, \mathbb{L})\), output ‘Failed’.
16: if \(UV^{-1}SW^{-1}\) is a scalar matrix then
18: Set \(\hat{D} = U \cdot V^{-1}\) and output \(\hat{D}\).
19: else
20: Output ‘Failed’. /* The check fails w.h.p if \(Z\) is not the ‘last’ matrix */
21: end if
Computing the kernels at random points (Steps 8–10): The following observation shows that the algorithm does not fail at step 9 with high probability. The proof is immediate from the above two observations and an application of the Schwartz-Zippel lemma.

**Observation 6.3** Let \( a_k, b_k \in \mathbb{F}^n \) for \( k \in [w + 1] \). Then, for every \( k \in [w + 1] \), and \( a = a_k \) or \( b_k \),

1. \( \operatorname{Ker}(Z'(a)) = \operatorname{span}_L \{(N_{11}(a) N_{12}(a) \ldots N_{1w}(a))^T \} \),

2. \( \operatorname{Ker}(Z'(a))^T = \operatorname{span}_L \{(N_{11}(a) N_{21}(a) \ldots N_{w1}(a))^T \} \),

and \( \operatorname{Ker}(F'_1(a_k)), \operatorname{Ker}(F'_1(b_k)) \) are one dimensional subspaces of \( \mathbb{L}^w \), with probability \( 1 - \frac{\lfloor wd \rfloor O(1)}{q} \).

Extracting \( D \) from the kernels (Steps 13–21): We analyse these steps for three separate cases. The analysis shows that if \( Z \) is the ‘last’ matrix then the algorithm succeeds with high probability, otherwise the test at step 17 fails with high probability.

**Case \([Z = C \cdot R \cdot D]\):** From Observation 6.2, \( \det(R'_j(a_k)) \) and \( \det(R'_j(b_k)) \) are nonzero with high probability, for all \( j \in [t - 1] \) and \( k \in [w + 1] \). Assuming this, the following holds for all \( k \in [w + 1] \):

\[
D \cdot \operatorname{Ker}(Z'(a_k)) = \operatorname{Ker}(F'_1(a_k))
\]

\[
D \cdot \operatorname{Ker}(Z'(b_k)) = \operatorname{Ker}(F'_1(b_k)).
\]

(6.2)

Hence, at step 9, there are \( \lambda_k, \rho_k \in \mathbb{L}^\times \) such that

\[
D \cdot v_k = \lambda_k u_k, \quad D \cdot s_k = \rho_k w_k \quad \text{for } k \in [w + 1].
\]

Step 13 also succeeds with high probability due to the following claim.

**Claim 6.3** With probability \( 1 - \frac{\lfloor wd \rfloor O(1)}{q} \), any subset of \( w \) vectors in any of the sets \( \{u_1, u_2, \ldots, u_{w+1}\}, \{v_1, v_2, \ldots, v_{w+1}\}, \{w_1, w_2, \ldots, w_{w+1}\} \), or \( \{s_1, s_2, \ldots, s_{w+1}\} \) are \( \mathbb{L} \)-linearly independent.

**Proof:** From Observation 6.3, for the sets \( \{v_1, v_2, \ldots, v_{w+1}\} \) and \( \{s_1, s_2, \ldots, s_{w+1}\} \) it is sufficient to show that any \( w \) columns of the \( w \times (w + 1) \) matrices \( (N_{i1}(a_j))_{i \in [w], j \in [w+1]} \) and \( (N_{i1}(b_j))_{i \in [w], j \in [w+1]} \) are \( \mathbb{L} \)-linearly independent with high probability. As the cofactors \( N_{11}, \ldots, N_{1w} \) are \( \mathbb{L} \)-linearly independent, the above follows from Claim 2.3. For the sets \( \{u_1, u_2, \ldots, u_{w+1}\} \) and \( \{w_1, w_2, \ldots, w_{w+1}\} \), it follows from Equation 6.2 that there are \( \lambda_k, \rho_k \in \mathbb{L}^\times \)
such that \( D \cdot v_k = \lambda_k u_k \) and \( D \cdot s_k = \rho_k w_k \) for all \( k \in [w+1] \). Since \( D \) is invertible, the claim follows for these two sets as well.

At this step, \( v_{w+1} = \sum_{k=1}^w \beta_k v_k \) and \( s_{w+1} = \sum_{k=1}^w \delta_k s_k \), and so by applying \( D \) on both sides,

\[
\lambda_{w+1} u_{w+1} = \sum_{k=1}^w \beta_k \lambda_k u_k, \quad \rho_{w+1} w_{w+1} = \sum_{k=1}^w \delta_k \rho_k w_k.
\]

Also, \( u_{w+1} = \sum_{k=1}^w \alpha_k u_k \) and \( w_{w+1} = \sum_{k=1}^w \gamma_k w_k \). By Claim 6.3, none of the \( \alpha_k, \beta_k, \gamma_k, \delta_k \) is zero and

\[
\frac{\lambda_k}{\lambda_{w+1}} = \frac{\alpha_k}{\beta_k}, \quad \frac{\rho_k}{\rho_{w+1}} = \frac{\gamma_k}{\delta_k}, \quad \text{for all } k \in [w].
\]

From the construction of matrices \( U, V, W \) and \( S \) at step 15,

\[
D \cdot V = \lambda_{w+1} U, \quad D \cdot S = \rho_{w+1} W,
\]

and \( U, V, W, S \in \text{GL}(w, L) \) (by Claim 6.3). Therefore, \( UV^{-1}SW^{-1} \) is a scalar matrix.

**Case b** \([Z^T = C \cdot R_t \cdot D] \): In this case, the check at step 17 fails with high probability. Suppose the algorithm passes steps 13 and 15, and reaches step 17. We show that \( UV^{-1}SW^{-1} \) being a scalar matrix implies an event \( \mathcal{E} \) that happens with a low probability. The event \( \mathcal{E} \) can be derived as follows:

Let \( M \stackrel{\text{def}}{=} U \cdot V^{-1} \), and \( c \in L^\times \) such that \( M = cW \cdot S^{-1} \). Assuming the invertibility of \( R'_j(a_k) \) and \( R'_j(b_k) \) for \( j \in [t-1] \) (Observation 6.2), and as in Equation 6.2, the following holds for all \( k \in [w+1] \).

\[
D \cdot \text{Ker}((Z'(a_k))^T) = \text{Ker}(F'_t(a_k)), \quad D \cdot \text{Ker}((Z'(b_k))^T) = \text{Ker}(F'_t(b_k)).
\]

By Observation 6.3, we can assume the above four kernels are one-dimensional. Hence, at step 9 there are \( p_k \in \text{Ker}((Z'(a_k))^T) \) and \( q_k \in \text{Ker}((Z'(b_k))^T) \) satisfying \( D \cdot p_k = u_k \) and \( D \cdot q_k = w_k \), for every \( k \in [w+1] \). Consider the \( w \times w \) matrices \( P \) and \( Q \) such that the \( k \)-th column of these matrices are \( \frac{\alpha_k}{\rho_k} p_k \) and \( \frac{\gamma_k}{\delta_k} q_k \), respectively, where \( \alpha_k, \beta_k, \gamma_k, \delta_k \) are the constants computed at step 13. Clearly, \( D \cdot P = U \) and \( D \cdot Q = W \), where \( U, W \) are the matrices computed at step 15.
As \( M = cW \cdot S^{-1} \) (by assumption), we have \( D^{-1}MS = cD^{-1}W = cQ \). Hence, for \( k \in [w] \),
\[
D^{-1}M \cdot s_k = \frac{c\gamma_k}{\delta_k} q_k.
\]

At step 13, \( w_w + 1 = \sum_{k=1}^{w} \gamma_k w_k \) and \( s_{w+1} = \sum_{k=1}^{w} \delta_k s_k \). Multiplying \( D^{-1} \) on both sides and \( D^{-1}M \) on both sides of these two equations respectively,
\[
q_{w+1} = \sum_{k=1}^{w} \gamma_k q_k, \quad \text{and} \quad D^{-1}M \cdot s_{w+1} = \sum_{k=1}^{w} c\gamma_k q_k.
\]
\[
\Rightarrow D^{-1}M \cdot s_{w+1} = c q_{w+1}.
\]

From Observation 6.3, there are \( \lambda_1, \lambda_2 \in \mathbb{L}^\times \) such that
\[
s_{w+1} = \lambda_1 \cdot (N_{11}(b_{w+1}) \cdot N_{12}(b_{w+1}) \ldots N_{1w}(b_{w+1}))^T, \quad q_{w+1} = \lambda_2 \cdot (N_{11}(b_{w+1}) \cdot N_{21}(b_{w+1}) \ldots N_{w1}(b_{w+1}))^T.
\]

Let \( D^{-1}M = (m_{lk})_{l,k \in [w]} \). Using the above values of \( s_{w+1} \) and \( q_{w+1} \) in Equation 6.3 and restricting to the first two entries of the resulting column vectors, we have
\[
\lambda_1 \left( \sum_{k=1}^{w} m_{1k} N_{1k}(b_{w+1}) \right) = c\lambda_2 N_{11}(b_{w+1}), \quad \lambda_1 \left( \sum_{k=1}^{w} m_{2k} N_{1k}(b_{w+1}) \right) = c\lambda_2 N_{21}(b_{w+1}).
\]

Thus we get the following relation,
\[
N_{21}(b_{w+1}) \left( \sum_{k=1}^{w} m_{1k} N_{1k}(b_{w+1}) \right) = N_{11}(b_{w+1}) \left( \sum_{k=1}^{w} m_{2k} N_{1k}(b_{w+1}) \right).
\]

Event \( E \) is defined by the above equality, i.e. we say \( E \) has happened whenever the above equality holds. Now observe that \( D^{-1}M \) is independent of the random bits used to choose \( b_{w+1} \), one way of seeing this is that \( D^{-1}M \) is already fixed before \( b_{w+1} \) is chosen. Hence, it is sufficient to show that the above equality happens with low probability over the randomness of \( b_{w+1} \), for any arbitrarily fixed \( m_{11}, \ldots, m_{1w} \) and \( m_{21}, \ldots, m_{2w} \) from \( \mathbb{L} \). Moreover, as \( D^{-1}M \) is invertible, we can assume – not all in \( \{m_{11}, \ldots, m_{1w}\} \) or \( \{m_{21}, \ldots, m_{2w}\} \) are zero. The following observation and Schwartz-Zippel lemma complete the proof in this case.

**Observation 6.4** \( N_{21}(z) (\sum_{k=1}^{w} m_{lk} \cdot N_{1k}(z)) \neq N_{11}(z) (\sum_{k=1}^{w} m_{lk} \cdot N_{1k}(z)) \) as polynomials in \( \mathbb{F}[z] \).
Proof: Suppose the two sides are equal. As $N_{21}(z)$ and $N_{11}(z)$ are irreducible and coprime polynomials, $N_{21}(z)$ must divide $\sum_{k=1}^{w} m_{2k} \cdot N_{1k}(z)$. But the two polynomials have the same degree and they are monomial disjoint, thereby giving us a contradiction. 

Case c $[Z = C \cdot R_i \cdot D \text{ or } Z^T = C \cdot R_i \cdot D$ for some $i \in [t - 1]]$: Assume $Z = C \cdot R_i \cdot D$ for some $i \in [t - 1]$. The case $Z^T = C \cdot R_i \cdot D$ can be argued similarly. Similar to Case b, we show that if the algorithm passes steps 13 and 15, and reaches step 17 then $UV^{-1}SW^{-1}$ being a scalar matrix implies an event $E$ that happens with very low probability. Hence, the check at step 17 fails with high probability. The event $E$ can be derived as follows:

Let $M \overset{\text{def}}{=} U \cdot V^{-1}$, and $c \in \mathbb{L}^\times$ be such that $M = c \cdot WS^{-1}$. From the construction of $W$ and $S$,

$$\frac{c \gamma_k}{\delta_k} w_k = M \cdot s_k, \quad \text{for all } k \in [w],$$

where $\gamma_k, \delta_k$ are as computed at step 13. Since $w_{w+1} = \sum_{k=1}^{w} \gamma_k w_k$ and $s_{w+1} = \sum_{k=1}^{w} \delta_k s_k$,

$$c \cdot w_{w+1} = M \cdot s_{w+1}. \quad (6.4)$$

Let $H' \overset{\text{def}}{=} D^{-1} \cdot R'_{t+1} \cdots R'_t$. From Observation 6.2, the following holds,

$$H^{-1} \cdot \text{Kernel}_{\mathbb{L}^\times}(Z') = \text{Kernel}_{\mathbb{L}^\times}(F'_t).$$

Let $n = (N_{11}(b_{w+1}) \quad N_{12}(b_{w+1}) \cdots N_{1w}(b_{w+1}))^T$. From Observation 6.3, and as $H(b_{w+1})$ is invertible with high probability over the random choice of $b_{w+1}$, there are $\lambda_1, \lambda_2 \in \mathbb{L}^\times$ such that

$$w_{w+1} = \lambda_1 H^{-1}(b_{w+1}) \cdot n$$

$$s_{w+1} = \lambda_2 n.$$

Substituting the above values of $w_{w+1}$ and $s_{w+1}$ in Equation 6.4, we have

$$c \lambda_1 H^{-1}(b_{w+1}) \cdot n = \lambda_2 M \cdot n, \quad \Rightarrow \quad c\lambda_1 n = \lambda_2 H(b_{w+1}) \cdot M \cdot n.$$

Let $H \cdot M = (h_{lk})_{l,k \in [w]}$. Restricting to the first two entries of the vectors in the above equality,
and observing that $M$ is independent of $b_{w+1}$, we have

$$c \lambda_1 N_{11}(b_{w+1}) = \lambda_2 \left( \sum_{k=1}^{w} h_{1k}(b_{w+1}) \cdot N_{1k}(b_{w+1}) \right),$$

$$c \lambda_1 N_{12}(b_{w+1}) = \lambda_2 \left( \sum_{k=1}^{w} h_{2k}(b_{w+1}) \cdot N_{1k}(b_{w+1}) \right).$$

Hence, we get the following relation

$$N_{11}(b_{w+1}) \cdot \left( \sum_{k=1}^{w} h_{2k}(b_{w+1}) \cdot N_{1k}(b_{w+1}) \right) = N_{12}(b_{w+1}) \cdot \left( \sum_{k=1}^{w} h_{1k}(b_{w+1}) \cdot N_{1k}(b_{w+1}) \right). \quad (6.5)$$

Event $E$ is defined by the above equality, that is $E$ happens if the above equality is satisfied. Observe that the entries of the matrix product $H \cdot M = (h_{lk})_{l,k \in [w]}$ are rational functions in $x$ variables and are independent of the random bits used to choose $b_{w+1}$. We show next the probability that the above equality holds is low over the randomness of $b_{w+1}$.

So far we have used only the first two properties of a pure product $F_i$, i.e, every $R_i$ is full-rank and $\det(R_1), \ldots, \det(R_t)$ are mutually coprime. However, these two properties are not sufficient to ensure the uniqueness of the last matrix in the product (as mentioned in a remark after Theorem 1.2). In the following observation, we use the third property of a pure product which ensures the desired uniqueness of the last matrix.

**Observation 6.5** Let $n \geq 2w^2$. Then all the entries of $H \cdot M$ are nonzero polynomials after setting the variables in $z_1 \overset{\text{def}}{=} \{z_{11}, z_{21}, z_{31}, \ldots, z_{w1}\}$ to zero.

**Proof:** $H \cdot M = D^{-1} \cdot R'_{t+1} \cdots R'_1 \cdot M = (h_{lk})_{l,k \in [w]}$. Recalling the substitution $z_{11} = \frac{-\sum_{k=2}^{w} z_{1k} N_{1k}}{N_{11}}$ at step 5, we observe that the rational function $h_{lk}$ becomes a polynomial under the setting $z_{11} = z_{21} = \ldots = z_{w1} = 0$, the variable $z_{11}$ does not even appear in $h_{lk}$. Let $Q_j = (R_j)_{z_1=0}$. By observing $(R_j)_{z_1=0} = (R_j')_{z_1=0}$, it follows that $(H \cdot M)_{z_1=0} = D^{-1} \cdot Q_{t+1} \cdots Q_1 \cdot M$. By the third property of a pure product, the entries of $Q_{t+1} \cdots Q_1$ are $\mathbb{L}$-linearly independent. Hence, none of the entries of $D^{-1} \cdot Q_{t+1} \cdots Q_1 \cdot M$ is zero, as $M \in \text{GL}(\mathbb{L}, w)$ whenever the algorithm passes step 15.

**Observation 6.6** $N_{11}(x) \cdot (\sum_{k=1}^{w} h_{2k}(x)N_{1k}(x)) \neq N_{12}(x) \cdot (\sum_{k=1}^{w} h_{1k}(x)N_{1k}(x))$ as rational functions in $\mathbb{L}(x)$.

**Proof:** Suppose $N_{11}(x) \cdot (\sum_{k=1}^{w} h_{2k}(x)N_{1k}(x)) = N_{12}(x) \cdot (\sum_{k=1}^{w} h_{1k}(x)N_{1k}(x))$. By substituting $z_1 = 0$ in the equation, the R.H.S becomes zero whereas the L.H.S reduces to $N_{11}^2 \cdot (h_{21})_{z_1=0}$,
which is nonzero (by Observation 6.5).

Noting that the degrees of the numerator and the denominator of \( h_{lk} \) are upper bounded by \( \omega d \), we conclude that the equality in Equation 6.5 happens with a low probability over the randomness of \( b_{w+1} \).

In case c if \( Z^T = C \cdot R_i \cdot D \) to begin with then the argument remains very similar except in Observation 6.5, the variables in the first row and column of \( Z \) (instead of just the first column) are substituted to zero.

6.4 Average-case ABP reconstruction: Proof of Theorem 1.3

The algorithm for average-case ABP reconstruction is presented in Algorithm 11, Section 6.1.1.2. The algorithm uses Algorithm 15 and Algorithm 16 during its execution – we present and analyze these two algorithms in the following subsections.

6.4.1 Computing the corner spaces

Let \( f \) be the polynomial computed by a random \((w,d,n)\)-ABP \( X_1 \cdot X_2 \cdots X_d \) over \( \mathbb{F} \), where \( n \geq 4w^2 \) and \( d \geq 5 \).

Lemma 6.2 With probability \( 1 - \frac{(\omega d)^{O(1)}}{q} \) over the randomness of \( f \), the following holds: Let \( K \supseteq \mathbb{F} \) be any field and \( f = 0 \mod \langle l_1, \ldots, l_k \rangle \), where the \( l_i \) are linear forms in \( K[x] \). Then \( k \geq w \) and for \( k = w \), the space \( \text{span}_K \{l_1, \ldots, l_w\} \) equals the \( K \)-span of either the linear forms in \( X_1 \) or the linear forms in \( X_d \).

The above uniqueness of the corner spaces, \( X_1 \) and \( X_d \) (defined in Section 6.1.1.2), helps compute them in Algorithm 15. The proof of the lemma is given at the end of this subsection.

Canonical bases of \( X_1 \) and \( X_d \): For a set of variables \( y \subseteq x \) and a linear form \( g \) in \( \mathbb{F}[x] \), define \( g(y) \overset{\text{def}}{=} g_{x\backslash y} \). We say \( g(y) \) is the linear form \( g \) projected to the \( y \) variables. Let \( x_1, \ldots, x_w \) and \( v \) be a designated set of \( w + 1 \) variables in \( x \), and \( u = x \setminus \{x_1, \ldots, x_w, v\} \). With \( n \geq 4w^2 \), a random \((w,d,n)\)-ABP \( X_1 \cdot X_2 \cdots X_d \) satisfies the following condition with probability \( 1 - \frac{w^{O(1)}}{q} \):

\[ (*) \text{a) The linear forms in } X_1 \text{ (similarly, } X_d \text{) projected to } x_1, \ldots, x_w \text{ are } \mathbb{F} \text{-linearly independent.} \]
If the above condition is satisfied then there is a $C \in \text{GL}(w, \mathbb{F})$ such that the linear forms in $X_1 \cdot C$ are of the kind:

$$x_i - \alpha_i v - g_i(u), \quad \text{for } i \in [w], \quad (6.6)$$

where each $\alpha_i \in \mathbb{F}$ and $g_i$ is a linear form in $\mathbb{F}[u]$. Thus, we can assume without loss of generality, the linear forms in $X_1$ are of the above kind. Similarly, the linear forms in $X_d$ are also of the kind:

$$x_i - \beta_i v - h_i(u), \quad \text{for } i \in [w], \quad (6.7)$$

where each $\beta_i \in \mathbb{F}$ and $h_i$ is a linear form in $\mathbb{F}[u]$. Moreover, with probability $1 - \frac{wD(1)}{q}$ over the randomness of the ABP, the following condition is satisfied:

$$(^b) \quad \alpha_1, \ldots, \alpha_w \text{ and } \beta_1, \ldots, \beta_w \text{ are distinct elements in } \mathbb{F}. \quad \text{(*)}$$

The task at hand for Algorithm 15 is to solve for $\alpha_i, g_i$ and $\beta_j, h_j$, for $i, j \in [w]$, assuming that conditions $(^a)$ and $(^b)$ are satisfied. The bases defined by Equations 6.6 and 6.7 are canonical for $X_1$ and $X_d$.

We analyze the three main steps of Algorithm 15 next:

1. **Partitioning the variables (Step 2):** The reason to partition the $u$ variables into sets $u_1, \ldots, u_m$ each of size $4w^2 - (w + 1)$ is to ensure that the polynomial $f_\ell = f_{u \setminus u_\ell} = 0$ for all $\ell \in [m]$ is computed by a random $(w, d, 4w^2)$-ABP. Hence the linear forms in the first, and the last matrix of the random ABP computing $f_\ell$ are $x_i - \alpha_i v - g_i(u_\ell)$ for $i \in [w]$, and $x_i - \beta_i v - h_i(u_\ell)$ for $i \in [w]$ respectively. The only thing to note here is, if $n - (w + 1)$ is not divisible by $4w^2 - (w + 1)$ then we allow the last two sets $u_{m-1}$ and $u_m$ to overlap – the algorithm can be suitably adjusted in this case.

2. **Reduction to solving systems of polynomial equations (Steps 5–13):** At step 7, the task of computing $(\alpha_1, \ldots, \alpha_w, g_1(u_\ell), \ldots, g_w(u_\ell))$ such that

$$f_\ell = 0 \mod \langle x_1 - \alpha_1 v - g_1(u_\ell), \ldots, x_w - \alpha_w v - g_w(u_\ell) \rangle,$$

can be reduced to solving for all $\mathbb{F}$-rational points of a system of polynomial equations over $\mathbb{F}$ as follows: Treat $\alpha_1, \ldots, \alpha_w$ and the $4w^3 - w(w+1)$ coefficients of $g_1(u_\ell), \ldots, g_w(u_\ell)$, say $w$, as formal variables. Substitute $x_i = \alpha_i v + g_i(u_\ell)$ for every $i \in [w]$ in the blackbox for $f_\ell$, and interpolate the resulting polynomial $p$ in the variables $\alpha_1, \ldots, \alpha_w, w, v, u_\ell$.

129
Algorithm 15 Computing the corner spaces

INPUT: Blackbox access to a $f$ computed by a random $(w, d, n)$-ABP.
OUTPUT: Bases of the two corner spaces $X_1$ and $X_d$ modulo which $f$ is zero.

1: /* Partitioning the variables */
2: Choose $w+1$ designated variables $x_1, x_2, \ldots, x_w, v$, and let $u = x \setminus \{x_1, \ldots, x_w, v\}$. Partition $u$ into sets $u_1, u_2, \ldots, u_m$, each of size $4w^2 - (w + 1)$.

3: /* Reduction to solving $m$ systems of polynomial equations */
4: for $\ell = 1$ to $m$ do
5: Set $f_\ell = f_{u \setminus u_\ell} = 0$.
6: Solve for all possible $(\alpha_1, \ldots, \alpha_w, g_1(u_\ell), \ldots, g_w(u_\ell))$, where each $\alpha_i \in \mathbb{F}$ and $g_i(u_\ell)$ is a linear form in $\mathbb{F}[u_\ell]$ such that $f_\ell = 0 \mod (x_1 - \alpha_1 v - g_1(u_\ell), \ldots, x_w - \alpha_w v - g_w(u_\ell))$.
7: if Step 6 does not return exactly two solutions for $(\alpha_1, \ldots, \alpha_w, g_1(u_\ell), \ldots, g_w(u_\ell))$ then
8: Output 'Failed'.
else
9: The solutions be $(\alpha_{\ell 1}, \ldots, \alpha_{\ell w}, g_1(u_\ell), \ldots, g_w(u_\ell))$ and $(\beta_{\ell 1}, \ldots, \beta_{\ell w}, h_1(u_\ell), \ldots, h_w(u_\ell))$.
10: end if

13: end for

14: /* Combining the solutions */
15: if $|\bigcup_{\ell \in [m]} \{(\alpha_{\ell 1}, \ldots, \alpha_{\ell w}), (\beta_{\ell 1}, \ldots, \beta_{\ell w})\}| \neq 2$ then
16: Output 'Failed'.
else
17: Without loss of generality, $(\alpha_{\ell 1}, \ldots, \alpha_{\ell w}) = (\alpha_1, \ldots, \alpha_w)$ and $(\beta_{\ell 1}, \ldots, \beta_{\ell w}) = (\beta_1, \ldots, \beta_w)$ for every $\ell \in [m]$. Set $g_i(u) = \sum_{\ell \in [w]} g_i(u_\ell)$ and $h_i(u) = \sum_{\ell \in [w]} h_i(u_\ell)$ for every $i \in [w]$.
18: Return $\{x_i - \alpha_i v - g_i(u)\}_{i \in [w]}$ and $\{x_i - \beta_i v - h_i(u)\}_{i \in [w]}$ as the bases of $X_1$ and $X_d$.
19: end if
with coefficients in \( \mathbb{F} \). The interpolation, which can be done in \( (d^w^3 \log q)^{O(1)} \) time, gives \( p \) in dense representation (i.e. as a sum of monomials). As the individual degrees of the variables in \( p \) are bounded by \( d \), we only need \( |\mathbb{F}| > d \) to carry out this interpolation. Now by treating \( p \) as a polynomial in the variables \( v, u_\ell \) with coefficients in \( \mathbb{F}(\alpha_1, \ldots, \alpha_w, w) \), and equating these coefficients to zero, we get a system of \( d^O(w^3) \) polynomial equations in \( O(w^3) \) variables with degree of each polynomial equation bounded by \( d \). By Lemma 6.2, such a system has exactly two solutions over \( \mathbb{F} \) and moreover, these two solution points are \( \mathbb{F} \)-rational. Hence, by applying Lemma 2.5, we can compute the two solutions for \( (\alpha_1, \ldots, \alpha_w, w) \) at step 7, in \( (d^w^3 \log q)^{O(1)} \) time.

3. Combining the solutions (Steps 16–21): The correctness of the steps follows from condition (**b).

Uniqueness of the corner spaces: Proof of Lemma 6.2

As \( n \geq 4w^2 \), a random \((w, d, n)\)-ABP \( X_1 \cdots X_d \) satisfies the following condition with probability \( 1 - \frac{(w^2 d)^{O(1)}}{q} \):

\[
  (**) \text{ For every choice of three (or less) matrices among } X_2, X_3, \ldots, X_{d-1}, \text{ the linear forms in these matrices and } X_1 \text{ and } X_d \text{ are } \mathbb{F} \text{-linearly independent.}
\]

So, it is sufficient to prove the following restatement of the lemma.

**Lemma 6.2.** Suppose \( f \) is computed by a \((w, d, n)\)-ABP \( X_1 \cdots X_d \) satisfying the above condition (**). If \( f = 0 \mod \langle l_1, \ldots, l_k \rangle \), where the \( l_i \) are linear forms over \( \mathbb{K} \supseteq \mathbb{F} \), then \( k \geq w \) and for \( k = w \), the space \( \text{span}_\mathbb{K}\{l_1, \ldots, l_w\} \) equals the \( \mathbb{K} \)-span of either the linear forms in \( X_1 \) or the linear forms in \( X_d \).

We prove the lemma first for \( d = 3 \), and then use this case to prove it for \( d > 3 \).

**Case \([d = 3]\):** There is an \( A \in \text{GL}(n, \mathbb{F}) \) such that \( f(Ax) \) is computed by \( (y_1 y_2 \ldots y_w) \cdot (r_{ij})_{i,j \in [w]} \cdot (z_1 z_2 \ldots z_w)^T \), where \( y = \{y_i\}_{i \in [w]}, \ r = \{r_{ij}\}_{i,j \in [w]} \) and \( z = \{z_j\}_{j \in [w]} \) are distinct variables in \( x \). If \( f = 0 \mod \langle l_1, \ldots, l_k \rangle \), then \( f(Ax) = 0 \mod \langle l_1(Ax), \ldots, l_k(Ax) \rangle \). Next, we show that if \( f(Ax) = 0 \mod k' \) linear forms \( h_1, \ldots, h_{k'} \in \mathbb{K}[y \cup z \cup r] \) then \( k' \geq w \), and for \( k' = w \), the space \( \text{span}_\mathbb{K}\{h_1, \ldots, h_w\} \) equals either \( \text{span}_\mathbb{K}\{y_1, \ldots, y_w\} \) or \( \text{span}_\mathbb{K}\{z_1, \ldots, z_w\} \). It follows that \( k \geq k' \geq w \), and for \( k = w \), the linear forms \( l_1(Ax), \ldots, l_w(Ax) \) must belong to \( \mathbb{K}[y \cup z \cup r] \) (otherwise, we will have \( f(Ax) = 0 \) modulo less than \( w \) linear forms in \( \mathbb{K}[y \cup z \cup r] \)).
and hence \( \text{span}_K \{ l_1, \ldots, l_w \} \) equals the \( K \)-span of either the linear forms in \( X_1 \) or the linear forms in \( X_d \).

Reusing symbols, assume that \( f \) is computed by \( X_1 \cdot X_2 \cdot X_3 \), where \( X_1 = (y_1 \ y_2 \ \ldots \ y_w) \), \( X_2 = (r_{ij})_{i,j \in [w]} \) and \( X_3 = (z_1 \ z_2 \ \ldots \ z_w)^T \), and \( f = 0 \mod \langle l_1, \ldots, l_k \rangle \), where the \( l_i \) are linear forms in \( K[y \cup z \cup r] \). Suppose \( k \leq w \); otherwise, we have nothing to prove. Consider the reduced Gröbner basis\(^1\) \( G \) of the ideal \( \langle l_1, \ldots, l_k \rangle \) with respect to the lexicographic monomial ordering defined by \( y \succ z \succ r \). There are sets \( S_y, S_z \subseteq [w] \) and \( S_r \subseteq [w] \times [w] \), satisfying \( |S_y| + |S_z| + |S_r| \leq k \), such that \( G \) consists of linear forms of the kind:

\[
\begin{align*}
y_i - g_i(y, z, r) & \quad \text{for } i \in S_y, \\
z_j - h_j(z, r) & \quad \text{for } j \in S_z, \\
r_{\ell e} - p_{\ell e}(r) & \quad \text{for } (\ell, e) \in S_r,
\end{align*}
\]

where \( g_i, h_j \) and \( p_{\ell e} \) are linear forms over \( K \) in their respective sets of variables. Let \( X'_1, X'_2, X'_3 \) be the linear matrices obtained from \( X_1, X_2, X_3 \) respectively, by replacing \( y_i \) by \( g_i(y, z, r) \), \( r_{\ell e} \) by \( p_{\ell e}(r) \) and \( z_j \) by \( h_j(z, r) \), for \( i \in S_y, (\ell, e) \in S_r \) and \( j \in S_z \). Then,

\[
X'_1 \cdot X'_2 \cdot X'_3 = 0. \tag{6.8}
\]

The dimension of the \( K \)-span of the linear forms of \( X'_1 \) is at least \( (w - |S_y|) \), that of \( X'_2 \) is at least \( (w^2 - |S_r|) \), and of \( X'_3 \) is at least \( (w - |S_z|) \). Also, there are \( C, D \in \text{GL}(w, K) \) such that \( X'_1 \cdot C, D \cdot X'_3 \) are obtained (via row and column operations on \( X'_1 \) and \( X'_3 \), respectively) from \( X_1, X_3 \) respectively, by replacing \( y_i \) by \( g_i(0, z, r) \) and \( z_j \) by \( h_j(0, r) \), for \( i \in S_y \) and \( j \in S_z \). Consider the following equation,

\[
(X'_1 C) \cdot (C^{-1} \cdot X'_2 D^{-1}) \cdot (DX'_3) = 0. \tag{6.9}
\]

By examining the L.H.S, we can conclude that for \( s \in [w] \setminus S_y \) and \( t \in [w] \setminus S_z \), the coefficient of the monomial \( y_s z_t \) over \( K(r) \) is the \( (s, t) \)-th entry of \( C^{-1} X'_2 D^{-1} \) which must be zero. Hence, the dimension of the \( K \)-span of the linear forms in \( C^{-1} X'_2 D^{-1} \) is at most \( w^2 - (w - |S_y|)(w - |S_z|) \). As the dimension of the \( K \)-span of the linear forms in \( X'_2 \) remains unaltered under left and

\(^1\)See [CLO07]. Equivalently, think of the set of linear forms obtained from a reduced row echelon form of the coefficient matrix of \( l_1, \ldots, l_k \).
right multiplications by elements in $GL(w, \mathbb{K})$, we get the relation

$$w^2 - |S_r| \leq w^2 - (w - |S_y|)(w - |S_z|)$$

$$\Rightarrow (w - |S_y|)(w - |S_z|) \leq |S_r|$$

$$\Rightarrow w^2 - (|S_y| + |S_z|)w + |S_y| \cdot |S_z| \leq |S_r|$$

$$\Rightarrow w^2 - (w - |S_r|)w + |S_y| \cdot |S_z| \leq |S_r|, \quad \text{as } |S_y| + |S_z| \leq k \leq w$$

$$\Rightarrow |S_r|w + |S_y| \cdot |S_z| \leq |S_r|.$$

As $|S_y|, |S_z|, |S_r| \geq 0$, we must have $|S_r| = 0$, and either $|S_y| = 0$ or $|S_z| = 0$.

Suppose $|S_r| = |S_z| = 0$ (the case for $|S_r| = |S_y| = 0$ is similar). Then, Equation 6.9 simplifies to

$$(X'_1C) \cdot (C^{-1}X_2) \cdot X_3 = 0.$$ 

If $k < w$ then there is a $y_s$ in $X_1$ that is not replaced while forming $X'_1$ from $X_1$. By examining the coefficient of $y_s$ over $\mathbb{K}(r, z)$ in the L.H.S of the above equation, we arrive at a contradiction. Hence, $k = w$, in which case Equation 6.8 simplifies to

$$X'_1 \cdot X_2 \cdot X_3 = 0.$$ 

The entries of $X'_1$ are linear forms in $z$ and $r$, and so $X'_1 = X'_1(z) + X'_1(r)$ where the entries of $X'_1(z)$ (similarly, $X'_1(r)$) are linear forms in $z$ (respectively, $r$). The above equation implies

$$X'_1(z) \cdot X_2 \cdot X_3 = 0 \quad \text{and} \quad X'_1(r) \cdot X_2 \cdot X_3 = 0,$$

as the two L.H.S above are monomial disjoint. It is now easy to argue that $X'_1(z) = X'_1(r) = 0$, implying $X'_1 = 0$ and hence the reduced Gröbner basis $G$ is in fact $\{y_1, \ldots, y_w\}$.

**Case** $[d > 3]$: As before, by applying an invertible transformation, we can assume that $X_1 = (y_1 \ y_2 \ \ldots \ \ y_w)$, $X_2 = (r_{ij})_{i,j \in [w]}$ and $X_d = (z_1 \ z_2 \ \ldots \ \ z_w)^T$. Let $u = x \setminus (y \cup z \cup r)$ and $k \leq w$. Consider the reduced Gröbner basis $G$ of the ideal $\langle I_1, I_2, \ldots, I_k \rangle$ with respect to the lexicographic monomial ordering defined by $u > y > z > r$. There are sets $S_u \subseteq [n - w^2 - 2w]$, $S_y, S_z \subseteq [w]$ and $S_r \subseteq [w^2]$, satisfying $|S_u| + |S_y| + |S_z| + |S_r| \leq k$, such that $G$ consists of linear forms of the kind:

$$u_m - t_m(u, y, z, r) \quad \text{for } m \in S_u,$$

133
\[y_i - g_i(y, z, r) \quad \text{for } i \in S_y,\]
\[z_j - h_j(z, r) \quad \text{for } j \in S_z,\]
\[r_{\ell e} - p_{\ell e}(r) \quad \text{for } (\ell, e) \in S_r,\]

where \(t_m, g_i, h_j\) and \(p_{\ell e}\) are linear forms over \(\mathbb{K}\) in their respective sets of variables. Let \(X'\) be the matrix obtained from \(X\) by replacing \(u_m\) by \(t_m(u, y, z, r)\), \(y_i\) by \(g_i(y, z, r)\), \(z_j\) by \(h_j(z, r)\), and \(r_{\ell e}\) by \(p_{\ell e}(r)\), for \(m \in S_u, i \in S_y, j \in S_z,\) and \((\ell, e) \in S_r\). Then,

\[X'_1 \cdot X'_2 \cdot X'_3 \cdots X'_d = 0.\]

Let \(X(u) \overset{\text{def}}{=} (X)_{y=z=r=0}\). By treating the L.H.S of the above equation as a polynomial in \(u\)-variables with coefficients from \(\mathbb{K}(y, z, r)\) and focusing on the degree-\((d - 3)\) homogeneous component of this polynomial, we have

\[X'_1 \cdot X'_2 \cdot X'_3(u) \cdots X'_{d-1}(u) \cdot X'_d = 0. \quad (6.10)\]

If \(X'_3(u) \cdots X'_{d-1}(u) \in \text{GL}(w, \mathbb{K}(u))\) then there is a \(c \in \mathbb{F}[u]\) such that \(C = X'_3(c) \cdots X'_{d-1}(c) \in \text{GL}(w, \mathbb{K})\). Define

\[f_1 = X_1 \cdot X_2 \cdot C \cdot X_d,\]

and observe that Equation 6.10 implies \(f_1\) is zero modulo the linear forms,

\[y_i - g_i(y, z, r) \quad \text{for } i \in S_y,\]
\[z_j - h_j(z, r) \quad \text{for } j \in S_z,\]
\[r_{\ell e} - p_{\ell e}(r) \quad \text{for } (\ell, e) \in S_r.\]

By applying Case \([d=3]\) on \(f_1\), we get the desired conclusion, i.e. \(k = w\) and the \(\mathbb{K}\)-span of the above linear forms (hence also that of \(\{l_1, \ldots, l_k\}\)) is either \(\text{span}_K\{y_1, \ldots, y_w\}\) or \(\text{span}_K\{z_1, \ldots, z_u\}\). So, suppose \(X'_3(u) \cdots X'_{d-1}(u) \notin \text{GL}(w, \mathbb{K}(u))\) in Equation 6.10. Then, there is a \(j \in [3, d - 1]\) such that \(\det(X'_j(u)) = 0\). Observe that \(X'_i(u)\) can be obtained from \(X_i(u)\) by replacing \(u_m\) by \(t_m(u, 0, 0, 0)\) for \(m \in S_u\). That is,

\[X'_i(u) = X_i(u) \mod \langle \{u_m - t_m(u, 0, 0, 0)\}_{m \in S_u} \rangle, \quad \text{for every } i \in [3, d - 1].\]

As \(X_j(u)\) is full-rank (which follows from condition (**)) and \(\det(X'_j(u)) = 0\), the fact below implies \(|S_u| = w, |S_y| = |S_z| = |S_r| = 0.\)
Observation 6.7 If the symbolic determinant $\text{Det}_w$ is zero modulo $s$ linear forms then $s \geq w$.

Hence, Equation 6.10 simplifies to

$$X_1 \cdot X_2 \cdot X_3'(u) \cdots X_{d-1}'(u) \cdot X_d = 0,$$

$$\Rightarrow X_3'(u) \cdots X_{d-1}'(u) = 0.$$  \hspace{1cm} (6.11)

The above equality can not happen and this can be argued by applying induction on the number of matrices in the L.H.S of Equation 6.11:

*Base case:* ($d = 4$) The L.H.S of Equation 6.11 has one matrix $X_3'(u)$. As $X_3(u)$ is full-rank (by condition (**)), it cannot vanish modulo $w$ linear forms.

*Induction hypothesis:* Equation 6.11 does not hold if the L.H.S has at most $d - 4$ matrices.

*Inductive step:* ($d > 4$) Suppose Equation 6.11 is true. As the $2w^2$ linear forms in $X_3(u)$ and $X_{d-1}(u)$ are linearly independent (condition (***) again), by Observation 6.7, at least one of $X_3'(u)$ and $X_{d-1}'(u)$ is invertible. This gives a shorter product where we can apply the induction hypothesis to get a contradiction.

6.4.2 Finding the coefficients in the intermediate matrices

Following the notations in Section 6.1.1.2, $y = \{y_1, \ldots, y_w\}$ and $z = \{z_1, \ldots, z_w\}$ are subsets of $x$, $r = x \setminus (y \oplus z)$, $X'_1 = (y_1 \ y_2 \ldots y_w)$ and $X'_d = (z_1 \ z_2 \ldots z_w)^T$. When Algorithm 11 reaches the third and final stage, it has blackbox access to a $f' \in \mathbb{F}[x]$ and linear matrices $S_2, \ldots, S_{d-1} \in \mathbb{L}[r]^{w \times w}$ returned by Algorithm 10, such that $S_2 \cdot S_3 \cdots S_{d-1}$ is the linear matrix factorization of a random $(w, d - 2, n - 2w)$-matrix product $R_2 \cdot R_3 \cdots R_{d-1}$ over $\mathbb{F}$. Further, there exist linear matrices $T_2, \ldots, T_{d-1} \in \mathbb{L}[x]^{w \times w}$ satisfying $(T_k)_{y=0, z=0} = S_k$ for every $k \in [2, d-1]$, such that $f'$ is computed by the ABP $X'_1 \cdot T_2 \cdots T_{d-1} \cdot X'_{d-1}$. The task for Algorithm 16 is to efficiently compute the coefficients of the $y$ and $z$ variables in $T_k$. At a high level, this is made possible because of the uniqueness of such $T_k$ matrices: Indeed the analysis of Algorithm 16 shows that with high probability the coefficients of $y$ and $z$ in $T_3, \ldots, T_{d-2}$ are uniquely determined, and (if a certain canonical form is assumed then) the same is true for matrices $T_2$ and $T_{d-1}$.

Canonical form for $T_2$ and $T_{d-1}$: Matrix $T_2$ is said to be in canonical form if for every $l \in [w]$ the coefficient of $y_l$ is zero in the linear form at the $(i, j)$-th entry of $T_2$, whenever $i > l$. Similarly, $T_{d-1}$ is in canonical form if for every $l \in [w]$ the coefficient of $z_l$ is zero in the linear form at the $(i, j)$-th entry of $T_{d-1}$ whenever $j > l$. It can be verified (see Section 4.2), if $f'$
is computed by an ABP \( X_1' \cdot T_2 \cdot \cdots \cdot T_{d-1} \cdot X_{d-1}' \) then it is computed by another ABP where the corresponding \( T_2 \) and \( T_{d-1} \) are in canonical form, and the other matrices remain unchanged.

Linear independence of minors of a random ABP: The lemma given below is the reason Algorithm 16 is able to reduce the task of finding the coefficients of the \( y \) and \( z \) variables to solving linear equations. In the following discussion, the \( i \)-th row and \( j \)-th column of a matrix \( M \) will be denoted by \( M(i, *) \) and \( M(\ast, j) \) respectively.

Let \( R_2 \cdot R_3 \cdot \cdots \cdot R_{d-1} \) be a random \((w, d - 2, n - 2w)\)-matrix product in \( r \)-variables over \( \mathbb{F} \). For every \( s, t \in [w] \), \( R_2(s, \ast) \cdot R_3 \cdot \cdots \cdot R_{d-2} \cdot R_{d-1}(\ast, t) \) is a random \((w, d - 2, n - 2w)\)-ABP having a total of \( w^2(d - 4) + 2w \) linear forms in all the \( R_k \) matrices. Let us index the linear forms arbitrarily by \([w^2(d - 4) + 2w]\). We associate a polynomial \( g_e^{(s,t)} \) with the \( e \)-th linear form, for every \( e \in [w^2(d - 4) + 2w] \), as follows: If the \( e \)-th linear form is the \((\ell, m)\)-th entry of \( R_k \) then

\[
g_e^{(s,t)}(r) \overset{\text{def}}{=} [R_2(s, \ast) \cdot R_3 \cdot \cdots \cdot R_{k-2} \cdot R_{k-1}(\ast, \ell)] \cdot [R_{k+1}(m, \ast) \cdot R_{k+2} \cdot \cdots \cdot R_{d-2} \cdot R_{d-1}(\ast, t)],
\]

by identifying the \( 1 \times 1 \) matrix of the R.H.S with the entry of the matrix. The polynomials \( \{g_e^{(s,t)} : e \in [w^2(d - 4) + 2w]\} \), will be called the minors of the ABP \( R_2(s, \ast) \cdot R_3 \cdot \cdots \cdot R_{d-2} \cdot R_{d-1}(\ast, t) \). Recall the iterated matrix multiplication polynomial (IMM\(_{w,d}\)) is computed by a \((w, d, n)\)-ABP \( X_1 \cdot X_2 \cdot \cdots \cdot X_d \) where each entry in \( X_i \) is a distinct variable for all \( i \in [d] \), and hence the set of minors of this ABP is equal to the set of first order partial derivatives of IMM\(_{w,d}\). This in spirit is similar to the more well-known notion of the minors of the \( \text{Det}_w \) polynomial which is equal to the set of its first order partial derivatives.

**Lemma 6.3** For \( d \geq 5 \), with probability \( 1 - \frac{(wd)^{O(1)}}{q} \) over the randomness of \( R_2 \cdot \cdots \cdot R_{d-1} \) the following holds: For every \( s, t \in [w] \), the minors \( \{g_e^{(s,t)} : e \in [w^2(d - 4) + 2w]\} \), are \( \mathbb{F} \)-linearly independent.

The proof of the lemma is given at the end of this section. Due to the uniqueness of factorization, the matrices \( S_2, \ldots, S_{d-1} \) in Algorithm 11 are related to \( R_2, \ldots, R_{d-1} \) as follows: There are \( C_1, D_1 \in \text{GL}(w, \mathbb{L}) \) such that \( S_i = C_i \cdot R_i \cdot D_i \), for every \( i \in [2, d - 1] \); moreover, there are \( c_2, \ldots, c_{d-2} \in \mathbb{L}^* \) satisfying \( C_2 = D_{d-1} = I_w \), \( D_i \cdot C_{i+1} = c_i I_w \) for \( i \in [2, d - 2] \), and \( \prod_{i=2}^{d-2} c_i = 1 \). Define minors of the ABP \( S_2(s, \ast) \cdot S_3 \cdot \cdots \cdot S_{d-2} \cdot S_{d-1}(\ast, t) \), for every \( s, t \in [w] \), like above. The edges of the ABP are indexed by \([w^2(d - 4) + 2w]\) and a polynomial \( h_e^{(s,t)} \) is associated with the \( e \)-th linear form as follows: If the \( e \)-th linear form is the \((\ell, m)\)-th entry of \( S_k \) then

\[
h_e^{(s,t)}(r) \overset{\text{def}}{=} [S_2(s, \ast) \cdot S_3 \cdot \cdots \cdot S_{k-2} \cdot S_{k-1}(\ast, \ell)] \cdot [S_{k+1}(m, \ast) \cdot S_{k+2} \cdots S_{d-2} \cdot S_{d-1}(\ast, t)]. \tag{6.12}
\]
It is a simple exercise to derive the following corollary from the lemma above.

**Corollary 6.1** For $d \geq 5$, with probability $1 - \frac{(wd)^{O(1)}}{q}$ the following holds: For every $s, t \in [w]$, the minors $\{h_e^{(s,t)} : e \in [w^2(d-4)+2w]\}$ are $L$-linearly independent.

We are now ready to argue the correctness of Algorithm 16 by tracing its steps.

1. **Computing the partial derivatives (Step 2):** In this step, we compute all the third order partial derivatives of $f'$ using Claim 2.2.

2. **Computing almost all the coefficients of the $y$ and $z$ variables (Steps 6–13):** Equations 6.13 and 6.14 are justified by treating $f'$ as a polynomial in the $y$ and $z$ variables with coefficients from $L(r)$, and examining the coefficients of $y_s^2z_t$ and $y_s z_t^2$ respectively. A linear system obtained at step 9 or step 11 has $w^2(d-4) + 2w$ variables and the same number of linear equations. Corollary 6.1, together with Claim 2.3, ensure that the square coefficient matrix of the linear system is invertible (with high probability), and hence the solution computed is unique. The uniqueness implies that the solutions obtained across multiple iterations of the loop do not conflict with each other. For instance, the coefficients of $y_s$ in the linear forms in $T_2(s, \ast), T_3, \ldots, T_{d-2}$ get computed repeatedly at step 9 for every value of $t \in [w]$ – uniqueness ensures that we always get the same values for these coefficients. This also shows that the matrices $T_3, \ldots, T_{d-2}$ are unique. By the end of this stage, the coefficients of $y$ and $z$ variables are computed for all the linear forms, except for the coefficients of $y_l$ in $T_2(s, \ast)$ for $l > s$, and the coefficients of $z_l$ in $T_{d-1}(\ast, t)$ for $l > t$. These coefficients are retrieved in the next stage.

3. **Computing the remaining $y$ and $z$ coefficients in $T_2$ and $T_{d-1}$ (Steps 16–19):** For an $s \in [w]$, consider the following minors of $S_2(s, \ast) \cdot S_3 \cdots S_{d-2} \cdot S_{d-1}(\ast, 1)$:

   $$S_3(m, \ast) \cdot S_4 \cdots S_{d-2} \cdot S_{d-1}(\ast, 1) \quad \text{for all } m \in [w].$$

   Without loss of generality, let these minors be $h_1^{(s,1)}, \ldots, h_w^{(s,1)}$. Let $l > s$. By treating $f'$ as a polynomial in the $y, z$ variables, with coefficients from $L(r)$, and examining the coefficient of $y_s y_l z_1$ in $f'$, we arrive at the equation,

   $$\sum_{e=1}^w c_e \cdot h_e^{(s,1)} + K(r) = \left( \frac{\partial f'}{\partial y_s y_l z_1} \right)_{y=0, z=0},$$

137
Algorithm 16 Computing the coefficients of $y$ and $z$ variables in $T_k$

INPUT: Blackbox access to $f'$ and linear matrices $S_2, \ldots, S_{d-1} \in \mathbb{L}[r]^{w \times w}$.
OUTPUT: Linear matrices $T_2, T_3, \ldots, T_{d-1} \in \mathbb{L}[x]^{w \times w}$ such that $f'$ is computed by $y \cdot T_2 \cdot T_3 \cdots T_{d-1} \cdot z^t$, satisfying $(T_k)_{y=0, z=0} = S_k$ for every $k \in [2, d-1]$.

1: /* Computing the partial derivatives */
2: Compute blackbox access to $(\frac{\partial f'}{\partial y_i y_j z^t})_{y=0, z=0}$ and $(\frac{\partial f'}{\partial y_i z^t})_{y=0, z=0}$ for all $s, l, t \in [w].$
3: For every $s, t \in [w]$, let $\{h_{c}(s,t) : c \in [w^2(d-4) + 2w]\}$ be the minors of the ABP $S_2(s,*) \cdot S_3 \cdots S_{d-2} \cdot S_{d-1}(*,t)$, as defined in Equation 6.12.
4:
5: /* Computing almost all the coefficients of the $y$ and $z$ variables in $T_k$ */
6: Set $E = w^2(d-4) + 2w$.
7: for every $s, t \in [w]$ do
8: Pick $a_1, \ldots, a_E \in_r \mathbb{F}[r]$ independently.
9: Solve the linear system over $\mathbb{L}$ defined by

$$\sum_{c \in [E]} c_c \cdot h_{c}(s,t) (a_i) = \left( \frac{\partial f'}{\partial y_i z^t} \right)_{y=0, z=0} (a_i), \quad \text{for } i \in [E],$$

(6.13)

for a unique solution of $\{c_c\}_{c \in [E]}$. If the coefficient matrix is not invertible, output 'Failed'.
10: For every $e \in [E]$, set the solution value of $c_e$ as the coefficient of $y_s$ in the $e$-th linear form of the ABP $T_2(s,*) \cdot T_3 \cdots T_{d-2} \cdot T_{d-1}(*,t)$.
11: Solve the linear system over $\mathbb{L}$ defined by

$$\sum_{e \in [E]} d_e \cdot h_{e}(s,t) (a_i) = \left( \frac{\partial f'}{\partial y_i z^t} \right)_{y=0, z=0} (a_i), \quad \text{for } i \in [E],$$

(6.14)

for a unique solution of $\{d_e\}_{e \in [E]}$. Set the solution value of $d_e$ as the coefficient of $z_t$ in the $e$-th linear form of the ABP $T_2(s,*) \cdot T_3 \cdots T_{d-2} \cdot T_{d-1}(*,t)$.
12: end for
13: end for
14:
15: /* Computing the remaining $y$ and $z$ coefficients in $T_2$ and $T_{d-1}$ */
16: for every $s, t \in [w]$ do
17: For every $l > s$, compute the coefficients of $y_l$ in the linear forms in $T_2(s,*)$ by setting up a linear system similar to Equation 6.13, but with the R.H.S replaced by $\frac{\partial f'}{\partial y_l y_j z^t}$.
18: For every $l > t$, compute the coefficients of $z_l$ in the linear forms in $T_{d-1}(*,t)$ by setting up a linear system similar to Equation 6.14, but with the R.H.S replaced by $\frac{\partial f'}{\partial y_l z^t}$.
19: end for
20:
21: The coefficients of the $r$ variables in the linear forms in $T_k$ remain the same as that in $S_k$, for all $k \in [2, d-1]$. Output $T_2, T_3, \ldots, T_{d-1}$.
where \( c_1, \ldots, c_w \) are the unknown coefficients of \( y_l \) in the linear forms of \( T_2(s,*) \), and \( K(r) \) is a known linear combination of some other minors. The fact that \( K(r) \) is known at step 17 follows from this observation – while forming a monomial \( y_s y_l z_1 \), we either choose \( y_s \) from \( X_1' \) and \( y_l \) from \( T_2(s,*) \) or \( T_3, \ldots, T_{d-1}(*,1) \), or \( y_l \) from \( X_1' \) and \( y_s \) from \( T_3, \ldots, T_{d-1}(*,1) \). In the latter case, we are using the fact that \( T_2 \) is in canonical form, and so \( y_s \) does not appear in \( T_2(l,*) \). As the coefficients of \( y_s, y_l \) in \( T_3, \ldots, T_{d-1}(*,1) \) are known from the computation in steps 6–13, we conclude that \( K(r) \) in known. Thus, we can solve for \( c_1, \ldots, c_w \) by plugging in \( w \) random points in place of the \( r \) variables and setting up a linear system in \( w \) variables. Corollary 6.1 and Claim 2.3 imply the \( w \times w \) coefficient matrix of the system is invertible, and hence the solution for \( c_1, \ldots, c_w \) is unique. The correctness of step 18 can be argued similarly, and this finally implies that \( T_2 \) and \( T_{d-1} \) (in canonical form) are unique.

**Linear independence of minors: Proof of Lemma 6.3**

We have to show that the minors of \( R_2(s,*) \cdot R_3 \cdots R_{d-2} \cdot R_{d-1}(*,t) \) are \( \mathbb{F} \)-linearly independent with high probability, for every \( s, t \in [w] \), where \( R_2 \cdot R_3 \cdots R_{d-1} \) is a random \((w,d-2,n-2w)\)-matrix product. We will prove it for a fixed \( s, t \in [w] \), and then by union bound the result will follow for every \( s, t \in [w] \). As \( n \geq 4w^2 \) and \( d \geq 5 \), we have \( n - 2w \geq 3w^2 \) and \( d - 2 \geq 3 \). So, it is sufficient to show the linear independence of the minors of a random \((w,d,n)\)-ABP \( X_1 \cdot X_2 \cdots X_d \) in \( x \)-variables, for \( n \geq 3w^2 \) and \( d \geq 3 \).

Treat the coefficients of the linear forms in \( X_1, \ldots, X_d \) as formal variables. In particular,

\[
X_1 = \sum_{i=1}^{n} U_i^{(1)} x_i, \quad X_k = \sum_{i=1}^{n} U_i^{(k)} x_i \quad \text{for} \quad k \in [2, d-1], \quad X_d = \sum_{i=1}^{n} U_i^{(d)} x_i, \tag{6.15}
\]

where \( U_i^{(1)} \) and \( U_i^{(d)} \) are row and column vectors of length \( w \) respectively, \( U_i^{(k)} \) is a \( w \times w \) matrix, and the entries of these matrices are distinct \( u \)-variables. We will denote the \((\ell, m)\)-th entry of \( U_i^{(k)} \) by \( U_i^{(k)}(\ell, m) \), and the \( m \)-th entry of \( U_i^{(d)} \) by \( U_i^{(d)}(m) \). From the above equations, \( X_1 \cdot X_2 \cdots X_d \) is a \((w,d,n)\)-ABP over \( \mathbb{F}(u) \). We will show in the following claim that the minors of this ABP are \( \mathbb{F}(u) \)-linearly independent. As the coefficients of the \( x \)-monomials of these minors are polynomials (in fact, multilinear polynomials) of degree \( d - 1 \) in the \( u \)-variables, an application of the Schwartz-Zippel lemma implies \( \mathbb{F} \)-linear independence of the minors (with high probability) when the \( u \)-variables are set randomly to elements in \( \mathbb{F} \) (as is done in a random ABP over \( \mathbb{F} \)).
Claim 6.4  The minors of \( X_1 \cdot X_2 \cdots X_d \) are \( \mathbb{F}(u) \)-linearly independent.

Proof:  We will prove by induction on \( d \).

Base case \((d=3)\): Clearly, if the minors are \( \mathbb{F} \)-linearly independent after setting the \( u \)-variables to some \( \mathbb{F} \)-elements then the minors are also \( \mathbb{F}(u) \)-linearly independent before the setting. As \( n \geq w^2 + 2w \), it is possible to set the \( u \)-variables in \( X_1, X_2, X_3 \) such that the entries of these matrices (after the setting) become distinct \( x \)-variables. The minors of this \( u \)-evaluated ABP \( X_1 \cdot X_2 \cdot X_3 \) are monomial disjoint and so \( \mathbb{F} \)-linearly independent.

Inductive step: Split the \( w^2(d-2) + 2w \) minors of \( X_1 \cdot X_2 \cdots X_d \) into two sets: The first set \( G_1 \) consists of minors \( g_e \) for \( e \in [w^2(d-3) + 2w] \), such that the \( e \)-th linear form is the \((\ell, m)\)-th entry of some matrix \( X_k \) satisfying \( k \neq d \) and if \( k = d - 1 \) then \( m = w \). The second set \( G_2 \) consists of minors \( g_e \) for \( e \in [w^2(d-3) + 2w + 1, w^2(d-2) + 2w] \), such that the \( e \)-th linear form is either the \((\ell, m)\)-th entry of \( X_{d-1} \) for \( m \neq w \), or the \( \ell \)-th entry of \( X_d \). Set \( G_1 \) has \( p = w^2(d-3) + 2w \) minors and \( G_2 \) has \( w^2 \) minors.

Suppose \( \mu_1, \ldots, \mu_p \) are monomials in \( x \)-variables of degree \( d-2 \). Imagine a \( (w^2(d-2) + 2w) \times (w^2(d-2) + 2w) \) matrix \( M \) whose rows are indexed by the minors in \( G_1 \) and \( G_2 \), and columns by monomials \( \mu_1 x_1, \mu_2 x_1, \ldots, \mu_p x_1 \) and \( x_2^{d-1}, x_3^{d-1}, \ldots, x_{w^2+1}^{d-1}, x_2, x_3, \ldots, x_{w^2+1} \). The \((g, \sigma)\)-th entry of \( M \) contains the coefficient of the monomial \( \sigma \) in \( g \), this coefficient is a multilinear polynomial in the \( u \)-variables. In a sequence of observations, we show that there exist \( \mu_1, \ldots, \mu_p \) such that \( \det(M) \neq 0 \).

Consider the variable \( u \equiv U_1^{(d)}(w) \). The following observations are easy to verify.

Observation 6.8  1. Variable \( u \) does not appear in any of the monomials of the \((g, \sigma)\)-th entry of \( M \) if \( g \in G_2 \) or \( \sigma \in \{x_2^{d-1}, \ldots, x_{w^2+1}^{d-1}\} \).

2. Variable \( u \) appears in some monomials of the \((g, \sigma)\)-th entry of \( M \) if \( g \in G_1 \) and \( \sigma \in \{\mu_1 x_1, \ldots, \mu_p x_1\} \), irrespective of \( \mu_1, \ldots, \mu_p \).

Observation 6.9  Let \( g \in G_1 \) and \( \sigma \in \{\mu_1 x_1, \ldots, \mu_p x_1\} \). If we treat the \((g, \sigma)\)-th entry of \( M \) as a polynomial in \( u \) with coefficients from \( \mathbb{F}[u \setminus u] \) then the coefficient of \( u \) does not depend on the variables:

(a) \( U_1^{(d)}(j) \) for \( j \neq w \) and \( i \in [n] \),

140
observation then follows from Claim 2.3.

Denote the union of the \( u \)-variables specified in (a), (b) and (c) of the above observation by \( v \).

**Observation 6.10** The set \( \{ g_{v=0} : g \in G_1 \} \) equals the set \( \{ h \cdot u x_1 : h \) is a minor of \( X_1 \cdot X_2 \cdots X_{d-1}(*,w) \} \).

By the induction hypothesis, the minors of \( X_1 \cdot X_2 \cdots X_{d-1}(*,w) \), say \( h_1, \ldots, h_p \), are \( \mathbb{F}(u) \)-linearly independent. Hence there are \( p \) monomials in \( x \)-variables of degree \( d - 2 \) such that \( h_1, \ldots, h_p \), when restricted to these monomials, are \( \mathbb{F}(u) \)-linearly independent. These \( p \) monomials are our choices for \( \mu_1, \ldots, \mu_p \). Let \( N \) be the \( p \times p \) matrix with rows indexed by \( h_1, \ldots, h_p \) and columns by \( \mu_1, \ldots, \mu_p \), and \( N(h, \mu) \) contains the coefficient of the monomial \( \mu \) in \( h \). Then, \( \det(N) \neq 0 \). Under these settings, we have the following observation (which can be derived easily from the above).

**Observation 6.11** The coefficient of \( u^p \) in \( \det(M) \), when treated as a polynomial in \( u \) with coefficients from \( \mathbb{F}[u \setminus u] \), is \( \det(N) \cdot \det(M_0) \), where \( M_0 \) is the submatrix of \( M \) defined by rows indexed by \( \{ g : g \in G_2 \} \) and columns by \( x_2^{d-1}, \ldots, x_{w^2+1}^{d-1} \).

The next observation completes the proof of the claim by showing \( \det(M) \neq 0 \).

**Observation 6.12** \( \det(M_0) \neq 0 \).

The proof of the above follows by noticing that \( M_0 \) looks like \( (f_i(u_j))_{i,j \in [w^2]} \), where \( u_1, \ldots, u_{w^2} \) are some disjoint subsets of the \( u \)-variables and \( f_1, \ldots, f_{w^2} \) are \( \mathbb{F} \)-linearly independent polynomials. The observation then follows from Claim 2.3.

### 6.4.3 Non-degenerate ABP

From the analysis, it can be easily shown that Theorem 1.3 gives a reconstruction algorithm for a \((w,d,n)\)-ABP \( X_1 \cdot X_2 \cdots X_d \), where \( 4w^2 \leq n \leq d^{w^2} \), and the following conditions are satisfied:

1. There are \( w + 1 \) variables \( \{ x_1, x_2, \ldots, x_w, v \} \subset X \) such that the linear forms in \( X_1 \) (similarly \( X_d \)) projected to \( x_1, x_2, \ldots, x_w \) (i.e. after setting the variables other than \( x_1, x_2, \ldots, x_w \) to zero) are \( \mathbb{F} \)-linearly independent. Further, if \( u = x \setminus \{ x_1, x_2, \ldots, x_w, v \} \) then in the bases \( \{ x_i - \alpha_i v - g_i(u) | i \in [w] \} \) and \( \{ x_i - \beta_i v - h_i(u) | i \in [w] \} \) of the spaces \( X_1 \) and \( X_d \) (defined in Section 6.1.1.2) respectively, where \( \alpha_i, \beta_i \in \mathbb{F} \) and \( g_i, h_i \) are linear forms in the \( u \)-variables, \( \alpha_1, \alpha_2, \ldots, \alpha_w, \beta_1, \beta_2, \ldots, \beta_w \) are distinct elements of \( \mathbb{F} \).
2. For every set \( r \subseteq x \) of size \( 4w^2 \) the following holds: The linear forms in \( X_1, X_d \) and every choice of three matrices among \( X_2, \ldots, X_{d-1} \), projected to the \( r \)-variables, are \( \mathbb{F} \)-linearly independent.

3. The matrix product \( X_2 \cdot X_3 \cdots X_{d-1} \) modulo the \( \mathbb{F} \)-linear space spanned by the linear forms in \( X_1 \) and \( X_d \) is a pure product.

4. The minors of the ABP \( X_2(s, \ast) \cdot X_3 \cdots X_{d-1}(\ast, t) \) (where \( X_2(s, \ast) \) denotes the \( s \)-th row of \( X_2 \) and \( X_{d-1}(\ast, t) \) the \( t \)-th column of \( X_{d-1} \)) modulo the \( \mathbb{F} \)-linear space spanned by the linear forms in \( X_1 \) and \( X_d \) are \( \mathbb{F} \)-linearly independent, for all \( s, t \in [w] \).

Given a \((w, d, n)\)-ABP, it can be checked whether the ABP satisfies condition 1 in deterministic \( \binom{n}{w+1}(wn \log q)^{O(1)} \) time, condition 2 in deterministic \( \binom{n}{4w^2}(wdn \log q)^{O(1)} \) time, and conditions 3 and 4 in randomized \( (wdn \log q)^{O(1)} \) time. The one thing to note here is that, to reconstruct an ABP satisfying the above conditions, Algorithm 15 needs to be slightly modified as follows: At step 2, instead of working with a designated set of \( w+1 \) variables, the algorithm checks condition 1 for every choice of \( w+1 \) variables till it finds a correct choice. Then the running time of the algorithm is \( \binom{n}{w+1}(wn \log q)^{O(1)} + (dw^3 n \log q)^{O(1)} \), which equals \( (dw^3 n \log q)^{O(1)} \) for \( n \leq dw^2 \).

### 6.5 Equivalence test for the determinant over finite fields

We prove Theorem 6.1 in this section. Using variable reduction (Algorithm 1) and translation equivalence (Algorithm 2) the determining whether \( f \) is an affine projection of \( \text{Det}_w \) via full-rank is reduced to equivalence testing. This is done in the same way as the reduction for \( \text{IMM} \) was done from Theorem 1.1a to Theorem 1.1b in Chapter 4. Thus for the rest of this section we set \( n = w^2 \) and determine whether the \( n \)-variate polynomial \( f \) is equivalent to \( \text{Det}_w \).

The equivalence test for \( \text{Det}_w \) is done in two steps: In the first step, the problem is reduced to the simpler problem of PS-equivalence testing. The second step then solves the PS-equivalence test. A \((w^2, w)\)-polynomial \( f \in \mathbb{L}[x] \) is PS-equivalent to \( \text{Det}_w \) if there is a permutation matrix \( P \) and a diagonal matrix \( S \in \text{GL}(w^2, \mathbb{L}) \) such that \( f = \text{Det}_w(PSx) \).

**Lemma 6.4 ([Kay12a])** There is a randomized algorithm that takes input blackbox access to \( f \), which is PS-equivalent to \( \text{Det}_w \), and with probability \( 1 - \frac{w^{O(1)}}{q} \) outputs a permutation matrix \( P \) and a diagonal matrix \( S \in \text{GL}(w^2, \mathbb{L}) \) such that \( f = \text{Det}_w(PSx) \). The algorithm runs in \((w \log q)^{O(1)}\) time.
It is in the first step where our algorithm differs from (and slightly simplifies) [Kay12a]. This reduction to PS-equivalence testing is given in Section 6.5.2. As in [Kay12a], the algorithm uses the structure of the group of symmetries and the Lie algebra of Det$_w$. An estimate of the probability that a random element of the Lie algebra of g$_{\text{Det}_w}$ has all its eigenvalues in $\mathbb{L}$ (Lemma 6.5) is key to the simplification in the first step.

### 6.5.1 Random element in the Lie algebra of determinant

From Lemma 3.7, it is easy to observe that g$_{\text{Det}_w}$ contains a diagonal matrix with distinct elements on the diagonal. The next claim can be proved using this observation.

**Claim 6.5** Let $L_1, \ldots, L_{2w^2-2}$ be an $\mathbb{F}$-basis of g$_{\text{Det}_w}$, and $L = \sum_{i=1}^{2w^2-2} \alpha_i \cdot L_i$, where $\alpha_1, \ldots, \alpha_{2w^2-2} \in \mathbb{F}$ are picked independently. Then, the characteristic polynomial of $L$ is square-free with probability $1 - \frac{w^{\Omega(1)}}{q}$.

The following lemma is the main technical contribution of this section. We use Lemma 6.5 together with Claim 2.1 to infer that with probability at least $\frac{1}{2w^2}$ a random matrix from the Lie algebra of a polynomial equivalent to Det$_w$ has $w^2$ distinct eigenvalues in $\mathbb{L}$, which crucially helps us to reduce equivalence testing to PS-equivalence testing.

**Lemma 6.5** Let $L_1, \ldots, L_{2w^2-2}$ be an $\mathbb{F}$-basis of g$_{\text{Det}_w}$, and $L = \sum_{i=1}^{2w^2-2} \alpha_i \cdot L_i$, where $\alpha_1, \ldots, \alpha_{2w^2-2} \in \mathbb{F}$ are picked independently. Then, the characteristic polynomial of $L$ is square-free and splits completely over $\mathbb{L}$ with probability at least $\frac{1}{2w^2}$.

**Proof:** Let $h(y)$ be the characteristic polynomial of $L$. From Claim 6.5, $h$ is square-free with probability $1 - \frac{w^{\Omega(1)}}{q}$. From Lemma 3.7, $L = L_1 + L_2$ where $L_1 \in L_{\text{row}}$ and $L_2 \in L_{\text{col}}$. As $L$ is uniformly distributed over g$_{\text{Det}_w}$, so is $L_1$ over $L_{\text{row}}$ and $L_2$ over $L_{\text{col}}$. In other words, if $L_1 = Z_1 \otimes I_w$ and $L_2 = I_w \otimes Z_2$ then $Z_1, Z_2$ are both uniformly (and independently) distributed over $Z_w$. If the characteristic polynomial of $Z_1$ (similarly $Z_2$) is irreducible over $\mathbb{F}$ then the eigenvalues of $Z_1$ (respectively, $Z_2$) lie in $\mathbb{L}$ and are distinct. If this happens for both $Z_1$ and $Z_2$ then there are $D_1, D_2 \in \text{GL}(w, \mathbb{L})$ such that $D_1^{-1}Z_1D_1$ and $D_2^{-1}Z_2D_2$ are diagonal matrices. This further implies,

$$(D_1^{-1} \otimes I_w) \cdot (I_w \otimes D_2^{-1}) \cdot L \cdot (I_w \otimes D_2) \cdot (D_1 \otimes I_w)$$

is a diagonal matrix, due to the observation below. In particular if $Z_1$ and $Z_2$ are diagonalizable then $L$ is diagonalizable.

**Observation 6.13** For any $M, N \in \mathbb{F}_{w \times w}$, $(M \otimes I_w)$ and $(I_w \otimes N)$ commutes. Also, if $M, N \in \text{GL}(w, \mathbb{F})$ then $(M \otimes I_w)^{-1} = (M^{-1} \otimes I_w)$ and $(I_w \otimes N)^{-1} = (I_w \otimes N^{-1})$.
We will show that the characteristic polynomial of \( Z \in_r \mathcal{Z}_w \) is irreducible with probability \( \delta \) and hence with probability at least \( \delta^2 \) the characteristic polynomial of \( L \) splits completely over \( \mathbb{L} \). Much like the proof of Claim 6.5, it can be shown that the characteristic polynomial of \( Z \in_r \mathcal{Z}_w \) is square-free with probability \( 1 - \frac{w^{O(1)}}{q} \). Hence, if the characteristic polynomial of \( Z \in_r \mathcal{Z}_w' \), where \( \mathcal{Z}_w' \subset \mathcal{Z}_w \) consists of matrices with distinct eigenvalues in \( \mathbb{F} \), is irreducible with probability \( \rho \) then \( \delta \geq \rho \cdot (1 - \frac{w^{O(1)}}{q}) \). Next, we lower bound \( \rho \).

Let \( \mathcal{P} \) be the set of monic, degree-\( w \), square-free polynomials in \( \mathbb{F}[y] \) with the coefficient of \( y^{w-1} \) equal to zero. Define a map \( \phi \) from \( \mathcal{Z}_w' \) to \( \mathcal{P} \),

\[
\phi : \ Z \mapsto \text{characteristic polynomial of } Z.
\]

Since \( Z \) is a traceless matrix the coefficient of \( y^{w-1} \) is zero in the characteristic polynomial of \( Z \). The map \( \phi \) is onto as the companion matrix of \( p(y) \in \mathcal{P} \) belongs to its pre-image under \( \phi \). Let \( \phi^{-1}(p(y)) \) be the set of matrices in \( \mathcal{Z}_w' \) that map to \( p \).

**Claim 6.6** Let \( p(y) \in \mathcal{P} \). Then

\[
\frac{(q^w - 1) \cdot (q^w - q) \cdots (q^w - q^{w-1})}{q^w} \leq |\phi^{-1}(p(y))| \leq \frac{(q^w - 1) \cdot (q^w - q) \cdots (q^w - q^{w-1})}{q^w(1 - \frac{w}{q})}.
\]

**Proof:** Let \( C_p \) be the companion matrix of \( p(y) \). If the characteristic polynomial of a \( Z \in \mathcal{Z}_w' \) equals \( p(y) \) then there is an \( E \in \text{GL}(w, \mathbb{F}) \) such that \( Z = E \cdot C_p \cdot E^{-1} \), as the eigenvalues of \( C_p \) are distinct in \( \mathbb{F} \). Moreover, for any \( E \in \text{GL}(w, \mathbb{F}) \), \( E \cdot C_p \cdot E^{-1} \in \mathcal{Z}_w' \) has characteristic polynomial \( p(y) \). Hence, \( \phi^{-1}(p(y)) = \{ E \cdot C_p \cdot E^{-1} \mid E \in \text{GL}(w, \mathbb{F}) \} \). Suppose \( E, F \in \text{GL}(w, \mathbb{F}) \) such that \( F \cdot C_p \cdot F^{-1} = E \cdot C_p \cdot E^{-1} \). Then \( E^{-1}F \) commutes with \( C_p \). Since \( C_p \) has distinct eigenvalues in \( \mathbb{F} \), \( E^{-1}F \) can be expressed as a polynomial in \( C_p \), say \( h(C_p) \), of degree at most \( w-1 \) with coefficients from \( \mathbb{F} \). Let \( \mathbb{F}[y]^{\leq (w-1)} \) denote the set of polynomials in \( \mathbb{F}[y] \) of degree at most \( w-1 \). Conversely, if \( h \in \mathbb{F}[y]^{\leq (w-1)} \) and \( h(C_p) \) is invertible then \( F = E \cdot h(C_p) \) is such that \( F \cdot C_p \cdot F^{-1} = E \cdot C_p \cdot E^{-1} \). As \( h_1(C_p) \neq h_2(C_p) \) for distinct \( h_1, h_2 \in \mathbb{F}[y]^{\leq (w-1)} \), we have

\[
|\phi^{-1}(p(y))| = \frac{|	ext{GL}(w, \mathbb{F})|}{|\{ h \in \mathbb{F}[y] : \deg(h) \leq (w - 1) \text{ and } h(C_p) \in \text{GL}(w, \mathbb{F}) \}|}.
\]

The numerator is exactly \( (q^w - 1) \cdot (q^w - q) \cdots (q^w - q^{w-1}) \), and the denominator is trivially upper bounded by \( q^w \). A lower bound on the denominator can be worked out as follows: Let \( \lambda_1, \ldots, \lambda_w \in \overline{\mathbb{F}} \) be the distinct eigenvalues of \( C_p \). If \( h(y) = a_{w-1}y^{w-1} + a_{w-2}y^{w-2} + \ldots + a_0 \in \mathbb{F}[y] \),
then \( h(\lambda_1), \ldots, h(\lambda_w) \) are the eigenvalues of \( h(C_p) \). Observe that
\[
\Pr_{h \in \mathbb{F}[y]^G} \{ h(\lambda_i) = 0, \text{ for some fixed } i \in [w] \} \leq \frac{1}{q},
\]
\[
\Rightarrow \Pr_{h \in \mathbb{F}[y]^G} \{ h(\lambda_i) = 0, \text{ for any } i \in [w] \} \leq \frac{w}{q},
\]
\[
\Rightarrow \Pr_{h \in \mathbb{F}[y]^G} \{ h(C_p) \in \text{GL}(w, \mathbb{F}) \} \geq 1 - \frac{w}{q}.
\]

Hence, the denominator is lower bounded by \( q^w(1 - \frac{w}{q}) \).

Let \( \rho_p = \frac{|G^{-1}(p(y))|}{|\mathbb{P}|} \), the probability that \( p(y) \) is the characteristic polynomial of \( Z \in_r \mathbb{Z}_w' \). From Claim 6.6, it follows that
\[
|\mathbb{Z}_w'| \leq \frac{(q^w - 1) \cdots (q^w - q)}{q^w(1 - \frac{w}{q})} \cdot |\mathbb{P}| \Rightarrow 1 - \frac{w}{q} \leq \rho_p \cdot |\mathbb{P}|.
\]

We show in the next claim that a \( p \in_r \mathbb{P} \) is irreducible over \( \mathbb{F} \) with probability at least \( \frac{1}{w}(1 - \frac{2}{q^{w/2}}) \), implying the characteristic polynomial of \( Z \in_r \mathbb{Z}_w' \) is irreducible over \( \mathbb{F} \) with probability \( \rho \geq \frac{1}{w}(1 - \frac{2}{q^{w/2}})(1 - \frac{w}{q}) \). Therefore, the probability that the characteristic polynomial of \( Z \in_r \mathbb{Z}_w \) is irreducible over \( \mathbb{F} \) is \( \delta \geq \frac{1}{w}(1 - \frac{2}{q^{w/2}})(1 - \frac{w}{q})(1 - \frac{w^2}{q}) \). As \( q \geq w^7 \), the probability that the characteristic polynomial of \( L \in_r \text{GL}_w \) splits completely over \( L \) is at least \( \delta^2 \geq \frac{1}{2w^2} \).

**Claim 6.7** A polynomial \( p \in_r \mathbb{P} \) is irreducible over \( \mathbb{F} \) with probability at least \( \frac{1}{w}(1 - \frac{2}{q^{w/2}}) \).

**Proof:** Let \( \mathcal{F} \) be the set of monic, degree-\( w \), square-free polynomials in \( \mathbb{F}[y] \). The difference between \( \mathcal{F} \) and \( \mathbb{P} \) is that a polynomial in \( \mathbb{P} \) additionally has coefficient of \( y^{w-1} \) equal to zero. We argue in the next paragraph that the fraction of \( \mathbb{F} \)-irreducible polynomials in \( \mathcal{F} \) and in \( \mathbb{P} \) are the same. As irreducible polynomials are square-free, the number of irreducible polynomials in \( \mathcal{F} \) is at least \( \frac{q^w - 2q^{w/2}}{w} \) \cite{vzGG03}. Hence, the fraction of irreducible polynomials in \( \mathcal{F} \) is at least \( \frac{1}{w}(1 - \frac{2}{q^{w/2}}) \).

Define a map \( \Psi \) from \( \mathcal{F} \) to \( \mathbb{P} \) as follows: For a \( u(y) = y^w + a_{w-1}y^{w-1} + \ldots + a_0 \in \mathcal{F} \), define \( \Psi(u) = u(y - \frac{a_{w-1}}{w}) \). Observe that the coefficient of \( y^{w-1} \) in \( \Psi(u) \) is zero. It is also an easy exercise to show that \( \Psi(u_1) = \Psi(u_2) \) if and only if there exists an \( a \in \mathbb{F} \) such that \( u_1(y) = u_2(y + a) \). As \( u(y) \) is irreducible over \( \mathbb{F} \) if and only if \( u(y + a) \) is irreducible over \( \mathbb{F} \), for \( a \in \mathbb{F} \), the fraction of \( \mathbb{F} \)-irreducible polynomials in \( \mathcal{F} \) is the same as that in \( \mathbb{P} \).

This completes the proof of Lemma 6.5.
6.5.2 Reduction to PS-equivalence testing

Algorithm 17 gives a reduction to PS-equivalence testing for $\text{Det}_w$. Suppose the input to the algorithm is a blackbox access to $f = \text{Det}_w(Ax)$, where $A \in \text{GL}(w^2, \mathbb{F})$. We argue the correctness of the algorithm by tracing its steps:

**Algorithm 17** Reduction to PS-equivalence

INPUT: Blackbox access to a $(w^2, w)$-polynomial $f \in \mathbb{F}[x]$ that is equivalent to $\text{Det}_w$ over $\mathbb{F}$.

OUTPUT: A $D \in \text{GL}(w^2, \mathbb{L})$ such that $f(Dx)$ is PS-equivalent to $\text{Det}_w$ over $\mathbb{L}$.

1: Compute an $\mathbb{F}$-basis of $g_f$. Let $\{F_1, F_2, \ldots, F_{2w^2-2}\}$ be the basis. Set $j = 1$.
2: for $j = 1$ to $w^2 \log q$ do
3: Pick $\alpha_1, \ldots, \alpha_{2w^2-2} \in \mathbb{F}$ independently. Set $F = \sum_{i \in [2w^2-2]} \alpha_i \cdot F_i$.
4: Compute the characteristic polynomial $h$ of $F$. Factorize $h$ into irreducible factors over $\mathbb{L}$.
5: if $h$ is square-free and splits completely over $\mathbb{L}$ then
6: Use the roots of $h$ to compute a $D \in \text{GL}(w^2, \mathbb{L})$ such that $D^{-1} \cdot F \cdot D$ is diagonal.
7: Exit loop.
8: else
9: Set $j = j + 1$.
10: end if
11: end for
12: if No $D$ found at step 7 in the loop then
13: Output ‘Failed’.
14: else
15: Output $D$.
16: end if

**Step 1:** An $\mathbb{F}$-basis of $g_f$ can be computed efficiently using Lemma 2.4.

**Step 3–12:** At step 4 an element $F$ of $g_f$ is chosen uniformly at random. By Claim 2.1, $F = A^{-1} \cdot L \cdot A$, where $L$ is a random element of $g_{\text{Det}_w}$. Lemma 6.5 implies, in every iteration of the loop, $h$ (at step 5) is square-free and splits completely over $\mathbb{L}$ with probability at least $\frac{1}{2w^2}$. Since the loop has $w^2 \log q$ iterations, the algorithm finds an $h$ that is square-free and splits completely over $\mathbb{L}$, with probability at least $1 - \frac{1}{q}$. Assume that the algorithm succeeds in finding such an $h$, and suppose $\lambda_1, \ldots, \lambda_{w^2} \in \mathbb{L}$ are the distinct roots of $h$. The algorithm finds a $D$ in step 7 by picking a random solution of the linear system obtained from the relation $F \cdot D = D \cdot \text{diag}(\lambda_1, \ldots, \lambda_{w^2})$ treating the entries of $D$ as formal variables. We argue next that
$f(Dx)$ is PS-equivalent to $\text{Det}_w$ over $\mathbb{L}$.

By Lemma 3.7, $L = L_1 + L_2$ where $L_1 \in \mathcal{L}_{\text{row}}$ and $L_2 \in \mathcal{L}_{\text{col}}$. In other words, there are $Z_1, Z_2 \in \mathbb{Z}_w$ such that $L_1 = Z_1 \otimes I_w$ and $L_2 = I_w \otimes Z_2$. It is easy to verify, if $L$ has distinct eigenvalues then so do $Z_1$ and $Z_2$. Hence, there are $D_1, D_2 \in \text{GL}(w, \mathbb{F})$ such that $D_1 Z_1 D_1^{-1}$ and $D_2 Z_2 D_2^{-1}$ are both diagonal, implying

$$M \overset{\text{def}}{=} (D_1 \otimes I_w) \cdot (I_w \otimes D_2) \cdot L \cdot (D_1^{-1} \otimes I_w) \cdot (I_w \otimes D_2^{-1})$$

is diagonal (by Observation 6.13) with distinct diagonal entries. Also,

$$D^{-1} \cdot F \cdot D = (AD)^{-1} \cdot L \cdot (AD) = ((D_1 \otimes I_w) \cdot (I_w \otimes D_2))^{-1} \cdot M \cdot ((D_1 \otimes I_w) \cdot (I_w \otimes D_2)) \cdot (AD)$$

As both $D^{-1} \cdot F \cdot D$ and $M$ are diagonal matrices with distinct diagonal entries, it must be that

$$(D_1 \otimes I_w) \cdot (I_w \otimes D_2) \cdot (AD) = P \cdot S,$$

where $P$ is a permutation matrix and $S \in \text{GL}(w^2, \mathbb{F})$ is a diagonal matrix (see Claim 63 of [Kay12a]). Now observe that $\text{Det}_w((D_1 \otimes I_w)x) = \beta \cdot \text{Det}_w(x)$ and $\text{Det}_w((I_w \otimes D_2)x) = \gamma \cdot \text{Det}_w(x)$, for $\beta, \gamma \in \mathbb{F} \setminus \{0\}$. Hence,

$$\text{Det}_w(P \cdot Sx) = \text{Det}_w((D_1 \otimes I_w) \cdot (I_w \otimes D_2) \cdot ADx) = \beta \gamma \cdot \text{Det}_w(ADx) = \beta \gamma \cdot f(Dx) \Rightarrow f(Dx) = \text{Det}_w(P \cdot S'x),$$

where $S' \in \text{GL}(w^2, \mathbb{F})$ is also diagonal. Therefore, $f(Dx)$ is $PS$-equivalent to $\text{Det}_w$ over $\mathbb{F}$. As $f(Dx) \in \mathbb{L}[x]$, it is a simple exercise to show that $f(Dx)$ must be $PS$-equivalent to $\text{Det}_w$ over $\mathbb{L}$.
Chapter 7

Lower bounds for IMM

In this chapter we present our lower bounds on multilinear and interval set-multilinear (ISM) formulas computing IMM. The contents of Section 7.2 are from our work [KNS16].

7.1 Introduction

Multilinear depth three formulas: An arithmetic formula is multilinear if every node in it computes a multilinear polynomial. Multilinear depth three and depth four circuits are defined in Definition 2.3. Note that at constant depth an arithmetic circuit can be transformed into an arithmetic formula with only a polynomial blow-up in size. The main result presented in this chapter is an exponential lower bound on the top fan-in of any multilinear depth three circuit computing IMM$_{w,d}$ (as defined in Section 2.3).

Theorem 1.4: Any multilinear depth three circuit (over any field) computing IMM$_{w,d}$ has top fan-in $w^{Ω(d)}$ for $w ≥ 6$.

Theorem 1.4 is proved in Section 7.2, and the main proof ideas are discussed in Section 7.1.1. After our work, [KST16b] proved a $w^{Ω(\sqrt{d})}$ lower bound on the size of multilinear depth four circuits computing IMM$_{w,d}$. As mentioned in Section 1.1, from the work of [Raz13] it follows that, to prove VBP $\neq$ VF it suffices to improve these results to either of the following:

a) a super-polynomial size lower bound on multilinear formulas computing IMM$_{w,d}$ where $d = O\left(\frac{\log w}{\log \log w}\right)$, or b) a $w^{Ω(\log d)}$ size lower bound on multilinear formulas computing IMM$_{w,d}$ for $d = w^{o(\log d')}$, where $d' = O\left(\frac{\log w}{\log \log w}\right)$. Implication (a) follows from [Raz13] easily, whereas for (b) the self-reducibility of IMM$_{w,d}$ can be used to show that a $w^{Ω(\log d)}$ size lower bound on multilinear formulas computing IMM$_{w,d}$ for $d = w^{o(\log d')}$, where $d' = O\left(\frac{\log w}{\log \log w}\right)$, implies a super-polynomial size lower bound on multilinear formulas computing IMM$_{w,d'}$. 148
As a consequence of the proof of Theorem 1.4 we also get an exponential separation between multilinear depth three and multilinear depth four circuits.

**Theorem 1.5:** There is an explicit family of $O(w^2d)$-variate polynomials of degree $d$, $\{f_d\}_{d \geq 1}$, such that $f_d$ is computable by a $O(w^2d)$-sized multilinear depth four circuit with top fan-in one (i.e. a $\Pi\Sigma\Pi$ circuit) and every multilinear depth three circuit computing $f_d$ has top fan-in $w^{\Omega(d)}$ for $w \geq 11$.

Theorems 1.4 and 1.5 are proved in Section 7.2. Theorem 1.4 implies the following corollary (already known due to [RY09]) as IMM$_{w,d}$ is a projection of Det$_{wd}$ that preserves multilinearity of the formula, i.e. under this projection every variable of IMM$_{w,d}$ has a unique pre-image consisting of a variable in Det$_{wd}$.

**Corollary 7.1 ([RY09])** Any multilinear depth three circuit (over any field) computing Det$_{d}$, the determinant of a $d \times d$ symbolic matrix, has top fan-in $2^{\Omega(d)}$.

**Interval set-multilinear (ISM) formulas:** Special kinds of multilinear formulas called ISM formulas were defined in Definition 1.2. It is well-known that if IMM$_{w,d}$ is computable by a size $s$ homogeneous non-commutative formula then there is a size $s^{O(1)}$ ISM formula computing IMM$_{w,d}$; we present a proof of this in Section 7.3.1. Thus to prove a super-polynomial size lower bound on homogeneous non-commutative formulas computing IMM$_{w,d}$ it is sufficient to prove a super-polynomial size lower bound on ISM formulas computing IMM$_{w,d}$. Such a lower bound would imply a super-polynomial separation between ABPs and homogeneous formulas in the non-commutative world, which is an important open problem. With the view of making progress on this problem, we consider an intermediate model—$\alpha$-balanced ISM formula (see Definition 1.3). We prove super-polynomial size lower bounds on $\alpha$-balanced ISM formulas computing IMM$_{w,d}$ when $\alpha = d^{-\frac{1}{\log d}}$, and $\alpha < \frac{1}{2}$.

**Theorem 1.7:** The size of any $\alpha$-balanced interval set-multilinear formula computing IMM$_{w,d}$ is $w^{\Omega\left(\frac{\log d}{\log \log d}\right)}$.

The proof of Theorem 1.7 is in Section 7.3.2, and the main proof ideas are discussed in Section 7.1.1. In Theorem 1.7 for $\alpha = d^{-\frac{1}{\log d}}$, where $f(d) = \omega(1)$ and $f(d) < \log d$, the depth of the formula is at least $\log d$ and at most $(d^{\frac{1}{\alpha}})^{-1} \log d$.

### 7.1.1 Proof strategy for Theorems 1.4 and 1.7

Theorem 1.4 is proved by introducing a new variant of the dimension of the space of partial derivatives measure called skewed partial derivatives, that is inspired by both [NW97] and
At a high level, the idea is to consider a polynomial $f$ in two sets of variables $x$ and $y$ such that $|y| \gg |x|$. If we take derivatives of $f$ with respect to all degree $k$ monomials in $y$ variables and set all the $y$ variables to zero after taking derivatives then we do expect to get a ‘large’ space of derivatives (especially, when $f$ is a ‘hard’ polynomial) simply because $|y|$ is large. On the other hand, in any depth three multilinear circuit $C$ computing $f$, the dimension of the space of derivatives of a product term is influenced only by the number of linear polynomials containing the $x$ variables as all the $y$ variables are set to zero subsequently. Thus, the measure is somewhat small for a product term of $C$ as $|x| \ll |y|$. By subadditivity of the measure (Lemma 2.1), this implies high top fan-in of $C$ computing $f$. A notable distinction from [Raz09, RY09] is that there is a significant difference in the sizes of the $|x|$ and $y$ variables (hence the name ‘skew’), and further, the variable sets $x$ and $y$ are fixed deterministically, a priori, and not by random partitioning of the entire set of variables.

Theorem 1.7 is proved in two steps. First, inspired by a depth-reduction result given for multilinear formulas in [Raz09], we show that if IMM$_{w,d}$ is computable by a size $s$ $\alpha$-balanced interval set-multilinear formula then it can be expressed as a sum of at most $s$ terms, where each term is a product of set-multilinear polynomials. Further, corresponding to each term $T$ in this expression there are $t = \Omega(\frac{\log d}{\log 2})$ indices $i_1, \ldots, i_t \in [d-1]$ such that the following holds: a) $1 \leq i_1 < i_2 < \ldots < i_t \leq d-1$, and b) $T = f_1 \cdots f_{t+1}$, where $f_1$ is a set-multilinear polynomial in the variable sets $x_1, \ldots, x_{i_1}$, $f_j$ is a set-multilinear polynomial in the variable sets $x_{i_j+1}, \ldots, x_{i_j}$ for $j \in [2, t]$, and $f_{t+1}$ is a set-multilinear polynomial in the variable sets $x_{i+1}, \ldots, x_d$. Additionally, we show that there is a set $S \subset [d]$, $|S| = t + 1$ such that for every term $T = f_1 \cdots f_{t+1}$, $f_j$ has variables from exactly one variable set $x_i$, for $j \in [t+1]$ and $i \in S$. In the second step, we substitute the $x$ variables as follows: a) for $i \notin S$ the variables in $x_i$ are substituted such that $Q_i$ turns into a $w \times w$ identity matrix after the substitution, that is $x_{u,v}^{(i)} = 0$ for all $u, v \in [w], u \neq v$, and $x_{u,u}^{(i)} = 1$ for all $u \in [w]$, b) the variables $x$ for $i \in S$ remain untouched. After such a substitution each term in the depth-reduction expression is transformed into a product of $t+1$ linear forms $\ell_1 \cdots \ell_{t+1}$, where $\ell_j$ has variables from exactly one variable set $x_i$, for $j \in [t+1]$ and $i \in S$. Hence, we have a multilinear depth three circuit (in fact set-multilinear depth three) with top fan-in at most $s$ that computes an IMM polynomial of width $w$ and length $t+1$. Since $t = \Omega(\frac{\log d}{\log 2})$, Theorem 1.4 implies $s = w^{\Omega(\frac{\log d}{\log 2})}$.

### 7.2 Lower bound on multilinear depth three formulas

The proofs of Theorems 1.4 and 1.5 are inspired by a particular kind of projection of IMM$_{w,d}$ considered in [FLMS15]. We say a polynomial $f$ is a simple projection of another polynomial
$g$ if $f$ is obtained by simply setting some variables to field constants in $g$.

### 7.2.1 Proof of Theorem 1.4

The proof proceeds by constructing an ABP $M$ of width $w$ and length $d$ such that (a) the polynomial computed by $M$, say $f$, is a simple projection of $\text{IMM}_{w,d}$, and (b) any multilinear depth three circuit computing $f$ has top fan-in $w^{\Omega(d)}$. Figure 1 depicts the ABP $M$.

![Figure 7.1: ABP $M$](image)

**Description of $M$.** The polynomial $f$, computed by $M$, is defined over two disjoint sets of variables, $x$ and $y$. The $y$ variables are contained in $k$ matrices, $\{Y_1, ..., Y_k\}$; the $(u, v)$-th entry in $Y_i$ is a formal variable $y^{(i)}_{u,v}$. There are $(k - 1)$ matrices $\{A_1, ..., A_{k-1}\}$, such that all the entries in these matrices are ones. The $x$ variables are contained in $2k$ matrices, $\{X_1, ..., X_{2k}\}$. Matrices $X_1$ and $X_{2k}$ are row and column vectors of size $w$ respectively. The $u$-th entry in $X_1$ (similarly, $X_{2k}$) is $x^{(1)}_u$ (respectively, $x^{(2k)}_u$). All the remaining matrices $\{X_2, ..., X_{2k-1}\}$ are diagonal matrices in the $x$ variables, i.e. the $(u, u)$-th entry in $X_i$ is $x^{(i)}_u$ and all other entries are zero for $i \in [2, 2k - 1]$. The matrices are placed as follows: Between two adjacent $Y$ matrices, $Y_i$ and $Y_{i+1}$, we have three matrices ordered from left to right as $X_{2i}, A_i$, and $X_{2i+1}$ for every $i \in [1, k - 1]$. Ordered from left to right, $X_1$ is on the left of $Y_1$ and $X_{2k}$ is on the right of $Y_k$. Naturally, we have the following relation among $k$ and $d$: $d = 4k - 1$, i.e. $k = \frac{d+1}{4}$. Thus $|x| = 2wk$ and $|y| = w^2k$. This imbalance between the number of $x$ and $y$ variables plays a vital role in the proof. Denote the polynomial computed by this ABP $M$ as $f(x, y)$.

The following claim is easy to verify as $f$ is a simple projection of $\text{IMM}_{w,d}$.

**Claim 7.1** If $\text{IMM}_{w,d}$ is computed by a multilinear depth three circuit having top fan-in $s$ then $f$ is also computed by a multilinear depth three circuit having top fan-in $s$.

We show every multilinear depth three circuit computing $f$ has top fan-in $w^{\Omega(d)}$ for $w \geq 6$. 

151
Lower bounding \( PD_{y_k}(f) \). Let \( \hat{Y}_k \subseteq Y_k \) be the set of monomials formed by picking exactly one \( y \)-variable from each of the matrices \( Y_1, \ldots, Y_k \) and taking their product. Then, \( |\hat{Y}_k| = w^{2k} \). Recall \( PD_{y_k}(f) \) denotes the skewed partial derivative of \( f \) as defined in Definition 2.13.

Claim 7.2 \( PD_{y_k}(f(x, y)) = |\hat{Y}_k| = w^{2k} \).

Proof: The derivative of \( f \) with respect to a monomial \( m \in Y_k \) is non-zero if and only if \( m \in \hat{Y}_k \). Also, such a derivative \( \frac{\partial f}{\partial m} \) is a multilinear degree-\( r \) monomial in \( x \)-variables. The derivatives of \( f \) with respect to two distinct monomials \( m \) and \( m' \) in \( \hat{Y}_k \) give two distinct multilinear degree-\( r \) monomials in \( x \)-variables. Hence, \( PD_{y_k}(f) = |\hat{Y}_k| \).

Upper bounding \( PD_{y_k} \) of a multilinear depth three circuit.

Lemma 7.1 Let \( C \) be a multilinear depth three circuit having top fan-in \( s \) computing a polynomial in \( x \) and \( y \) variables. Then \( PD_{y_k}(C) \leq s \cdot (k + 1) \cdot \binom{|x|}{k} \) if \( k \leq \frac{|x|}{2} \).

Proof: Let \( C = \sum_{i=1}^s T_i \), where each \( T_i \) is a product of linear polynomials on disjoint sets of variables. From Lemma 2.1, \( PD_{y_k}(C) \leq s \cdot \max_{i \in [s]} PD_{y_k}(T_i) \). We need to upper bound the dimension of the “skewed” partial derivatives of a term \( T_i = T \) (say). Let \( T = \prod_{j=1}^q l_j \), where \( l_j \) is a linear polynomial. Among the \( q \) linear polynomials at most \( |x| \) of them contain the \( x \)-variables. Without loss of generality, assume the linear polynomials \( l_1, \ldots, l_p \) contain \( x \)-variables and the remaining \( l_{p+1}, \ldots, l_q \) are \( x \)-free (here \( p \leq |x| \)). Let \( Q = \prod_{j=p+1}^q l_j \). Then, \( T = Q \cdot \prod_{j=1}^p l_j \). We take the derivative of \( T \) with respect to a monomial \( m \in Y_k \) and then substitute the \( y \) variables to zero. Applying the product rule of differentiation and observing that the derivative of a linear polynomial with respect to a variable makes it a constant we have the following:

\[
\left[ \frac{\partial T}{\partial m} \right]_{y=\bar{y}} = \sum_{S \subseteq \{p\} \atop |S| \leq k} \alpha_S \prod_{j \in \{p\} \setminus S} [l_j]_{y=\bar{y}}
\]

where \( \alpha_S \)'s are constants from the field. Here \( m \) is a representative element of the set \( Y_k \). Hence every such derivative can be expressed as a linear combination of \( \sum_{t=0}^k \binom{p}{t} \leq (k + 1) \cdot \binom{|x|}{k} \) polynomials, where the last inequality is due to \( k \leq \frac{|x|}{2} \) (if \( t > p \) then \( \binom{p}{t} \) def \( = 0 \)). Therefore, \( PD_{y_k}(T) \leq (k + 1) \cdot \binom{|x|}{k} \) and \( PD_{y_k}(C) \leq s \cdot (k + 1) \cdot \binom{|x|}{k} \).

It follows from Claim 7.2 and Lemma 7.1 that the top fan-in \( s \) of any multilinear depth three circuit computing \( f(x, y) \) is such that

\[
s \geq \frac{w^{2k}}{(k + 1) \cdot \binom{|x|}{k}} \geq \frac{w^{2k}}{(k + 1) \cdot (2we)^k} = w^{\Omega(d)},
\]

152
as \( w \geq 6 \) and \( k \leq |x|/2 \) (required in Lemma 7.1). Claim 7.1 now completes the proof of Theorem 1.4. ■

### 7.2.2 Proof of Theorem 1.5

We now show that the polynomial \( f(x, y) \), computed by the ABP \( M \), can also be computed a multilinear depth four circuit of size \( O(w^2d) \) and having top fan-in just one. ABP \( M \) has \( k \) matrices, \( Y_1, \ldots, Y_k \), containing the \( y \)-variables. Associate with each matrix \( Y_i \) two matrices containing the \( x \) variables, one on the immediate left \( X_{2i-1} \), and one on the immediate right \( X_{2i} \). Every monomial in \( f \) is formed by picking exactly one variable from every matrix and taking their product. Once we pick \( y_{u,v}^{(i)} \) from \( Y_i \), this automatically fixes the variables picked from \( X_{2i-1} \), and \( X_{2i} \), as these are diagonal matrices. Moreover, any variable can be picked from \( Y_i \) irrespective of which other \( y \) variables are picked from \( Y_1, \ldots, Y_{i-1}, Y_{i+1}, \ldots, Y_k \). This observation can be easily formalized to show that

\[
f = \prod_{i=1}^{k} \sum_{u,v \in [n]} x_{u}^{(2i-1)} \cdot y_{u,v}^{(i)} \cdot x_{v}^{(2i)}.
\]

The size of this multilinear \( \Pi \Sigma \Pi \) circuit is \( O(w^2k) = O(w^2d) \). ■

### 7.3 Lower bound on \( \alpha \)-balanced ISM formulas

#### 7.3.1 Reduction from homogeneous non-commutative formulas to ISM formulas

The variables in a non-commutative polynomial do not commute under multiplication operation. For example \( x_1x_2 - x_2x_1 \) is a non-zero non-commutative polynomial. A homogeneous non-commutative formula is defined akin to a homogeneous commutative formula, i.e. it is an arithmetic formula where every node in it computes an homogeneous non-commutative polynomial with the only exception being that there is now an ordering among the children of the product gates to determine the order of multiplication. In particular, if \( v \) is a product node in a homogeneous non-commutative formula with children \( v_1, \ldots, v_k \) ordered from left to right as \( v_1 \succ v_2 \succ \ldots \succ v_k \), and suppose \( \phi_v, \phi_{v_i} \) denotes the non-commutative polynomial computed by nodes \( v, v_i \) respectively for all \( i \in [k] \), then \( \phi_v = \phi_{v_1} \cdot \phi_{v_2} \cdots \phi_{v_k} \). In Lemma 7.2 we prove that if \( \text{IMM}_{w,d} \) is computed by a size \( s \) fan-in two depth \( d \) homogeneous non-commutative formula then it is also computed by a size \( s \) depth \( d \) ISM formula. Since a size \( s \) homogeneous non-commutative formula (of arbitrary fan-in) can be easily transformed into
Lemma 7.2 implies that if \( \text{IMM}_{w,d} \) is computed by a size \( s \) homogeneous non-commutative formula (of arbitrary fan-in) then it is also computed by a size \( s^{O(1)} \) ISM formula. In case of fan-in two homogeneous non-commutative formula, we refer to the children \( v_1, v_2 \) of a product node \( v \) as its left and right child respectively with \( v_1 \succ v_2 \).

Lemma 7.2 If \( \text{IMM}_{w,d} \) is computed by a size \( s \) depth \( d \) fan-in two homogeneous non-commutative formula then it is also computed by a size \( s \) depth \( d \) interval set-multilinear formula.

Proof: Let \( \varphi \) be a fan-in two homogeneous non-commutative formula computing \( \text{IMM}_{w,d} \). Since the formula is homogeneous we assume without loss of generality that every leaf node is labelled by a variable and hence every node in \( \varphi \) computes at least a degree one polynomial.

We associate with every node in \( \varphi \) an interval \( I \subseteq [1, d] \) such that the degree of the polynomial computed by a node \( v \) in \( \varphi \) is equal to the length of the interval associated with it. The intervals are associated inductively starting from the root, which is assigned the interval \([1, d]\). Since the root of \( \varphi \) computes \( \text{IMM}_{w,d} \), the length of the interval associated with it is equal to the degree of the polynomial it computes. Assume inductively we have assigned an interval \( I = [i_1, i_2] \), where \( 1 \leq i_1 \leq i_2 \leq d \) with a node \( v \) computing a polynomial of degree \( d_v \) equal to \( |I| = i_2 - i_1 + 1 \). The intervals associated with the children of \( v \) are determined as follows:

1. If \( v \) is an addition node with children \( v_1 \) and \( v_2 \) then we associate the same interval \( I \) with both the children \( v_1 \) and \( v_2 \). Since \( \varphi \) is homogeneous, the degree of the polynomials computed by \( v_1 \) and \( v_2 \), denoted \( d_{v_1} \) and \( d_{v_2} \) respectively, is equal to \( d_v \) the degree of its parent node.

2. Suppose \( v \) is a product node and \( v_1 \) and \( v_2 \) is its left and right child respectively. Since \( \varphi \) is homogeneous, the degrees of the polynomials computed by nodes \( v_1 \) and \( v_2 \), denoted \( d_{v_1} \) and \( d_{v_2} \) respectively, satisfy \( d_{v_1} + d_{v_2} = d_v \) and \( d_{v_1}, d_{v_2} > 0 \). Let \( i_3 = i_1 + d_{v_1} - 1 \). Then associate the intervals \([i_1, i_3]\) and \([i_3+1, i_2]\) with nodes \( v_1 \) and \( v_2 \) respectively. It is easy to infer that the degrees of the polynomials computed by the nodes \( v_1 \) and \( v_2 \) are equal to the intervals associated with it.

Since the leaf nodes compute degree one polynomials, they are associated with unit length intervals of the type \([i, i]\), where \( i \in [d] \). Perform the following substitution at every leaf node: if a leaf node \( v \) associated with an interval \([i, i]\) is labelled by a variable \( x \in x_j \), where \( i, j \in [d] \) and \( i \neq j \), then substitute the leaf node by 0. Call the formula \( \varphi \) after this substitution as \( \psi \). It is easy to verify that \( \psi \) along with the intervals associated with its nodes as above is an
interval set-multilinear formula in variable sets $x_1, \ldots, x_d$, that is if $I$ is the interval associated with a node $v$ then $v$ computes a set-multilinear polynomial in the variables $x_i$, $i \in I$. Next we argue $\psi$ still computes $\text{IMM}_{w,d}$.

Let $v$ be a node in $\varphi$, $\varphi_v$ be the degree $d_v$ polynomial computed by $v$, and $I = [i_1, i_2]$ be the interval associated with $v$ as above. Then $d_v = i_2 - i_1 + 1$. Partition the degree $d_v$ non-commutative monomials in $x_1, \ldots, x_d$ variables that have non-zero coefficient in $\varphi_v$ into valid and invalid monomials as follows: a monomial $\mu = x_{j_1}x_{j_2} \cdots x_{j_{d_v}}$ is valid if $x_{j_k} \in x_{i_{1+k-1}}$ for all $k \in [d]$, and the remaining monomials are invalid. Write $\varphi_v$ as a sum of two polynomials $\varphi_v,\text{valid}$ and $\varphi_v,\text{invalid}$. Every monomial in $\varphi_v,\text{valid}$ (respectively $\varphi_v,\text{invalid}$) is a valid (respectively invalid) monomial. The following claim completes the proof of the lemma.

**Claim 7.3** Let $v$ be a node in $\varphi$ and $\psi$, and $\varphi_v,\psi_v$ be the polynomials computed by $v$ in $\varphi,\psi$ respectively. Then $\varphi_v,\text{valid} = \psi_v$.

**Proof:** We prove this claim using induction starting from the leaves. Let $v$ be a leaf node labelled by $x$, and $[i, i]$ be the interval associated with $v$, where $i \in [1, d]$. If $x \in x_i$ then $\varphi_v = \varphi_v,\text{valid} = \psi_v$, and if $x \notin x_i$ then $\varphi_v,\text{valid} = \psi_v = 0$. Hence the claim is true for all leaf nodes. Let $v_1$ and $v_2$ be the children of $v$ in $\varphi$ and $\psi$, and $\varphi_{v_1},\psi_{v_1}$ (respectively $\varphi_{v_2},\psi_{v_2}$) be the polynomials computed by $v_1$ (respectively $v_2$) in $\varphi,\psi$ respectively. Then from induction hypothesis, $\varphi_{v_1,}\text{valid} = \psi_{v_1}$ and $\varphi_{v_2,}\text{valid} = \psi_{v_2}$. If $v$ is an addition node then the same intervals are associated with $v, v_1,$ and $v_2$. Hence, $\varphi_v,\text{valid} = \varphi_{v_1,}\text{valid} + \varphi_{v_2,}\text{valid} = \psi_{v_1} + \psi_{v_2} = \psi_v$. If $v$ is a product node then observe that $\varphi_v,\text{valid} = \varphi_{v_1,}\text{valid} \cdot \varphi_{v_2,}\text{valid} = \psi_{v_1} \cdot \psi_{v_2}$.

Let $r$ be the root of $\varphi$, $\psi$, and $\varphi_r,\psi_r$ be the polynomials computed by $r$ in $\varphi,\psi$ respectively. Then from Claim 7.3, $\varphi_r = \varphi_r,\text{valid} = \psi_r = \text{IMM}_{w,d}$.

**7.3.2 Proof of Theorem 1.7**

Recall $\text{IMM}_{w,d}$ is the entry of the $1 \times 1$ matrix product $Q_1 \cdots Q_t$ as defined in Section 2.3. Let $\varphi$ be a size $s$, $\alpha$-balanced ISM formula computing $\text{IMM}_{w,d}$. Let $t$ be such that $\lfloor \alpha^t d \rfloor \geq 2$, and $\lfloor \alpha^{t+1} d \rfloor < 2$, and $S = \{1, 1 + \lfloor (1 - \alpha)d \rfloor, 1 + \lfloor (1 - \alpha^2)d \rfloor, \ldots, 1 + \lfloor (1 - \alpha^t)d \rfloor\}$. Notice that for $i < t$ and $0 < \alpha < \frac{1}{2}$, $(1 - \alpha^i)d + 1 < (1 - \alpha^{i+1})d$. Hence, $|S| = t + 1$. For convenience of notation we index the elements in $S$ as follows: $S[1] = 1$, and $S[a] = 1 + \lfloor (1 - \alpha^{a-1})d \rfloor$ for $a \in [2, t + 1]$.

Let $g_1, \ldots, g_m$ be $m$ set-multilinear polynomials and $i_1, \ldots, i_{m-1} \in [d - 1]$, where $1 \leq i_1 < i_2 < \ldots < i_{m-1} \leq d - 1$, such that $g_1$ is a set-multilinear polynomial in the variable sets $x_1, \ldots, x_{i_1}$, 155
$g_a$ is a set-multilinear polynomial in the variable sets $x_{i_a-1+1},\ldots,x_{i_a}$ for $a \in [2,m-1]$, and $g_m$ is a set-multilinear polynomial in the variable sets $x_{i_{m-1}+1},\ldots,x_d$. Then we say the product $g_1 \cdots g_m$ is a $m$-set-multilinear product with respect to the indices $(i_1,\ldots,i_{m-1})$. In Lemma 7.3, we show that $\text{IMM}_{w,d}$ can be expressed as a sum of at most $s$ (the size of $\varphi$) terms which are $(t+1)$-set-multilinear products, and satisfy a certain property with respect to set $S$.

**Lemma 7.3** If $\varphi$ is a size $s$, $\alpha$-balanced interval set-multilinear formula computing $\text{IMM}_{w,d}$ then $\text{IMM}_{w,d}$ can be expressed as sum of at most $s$ terms, where each term is equal to the product of $t+1$ set-multilinear polynomials. Further, corresponding to each term $T_p = f_{p,1} \cdots f_{p,t+1}$ $p \in [s]$, there are $t$ indices $1 \leq i_{p,1} < i_{p,2} < \ldots < i_{p,t} \leq d-1$ such that the following two conditions are satisfied:

1. $f_{p,1} \cdots f_{p,t+1}$ is a $(t+1)$-set-multilinear product with respect to the indices $(i_{p,1},\ldots,i_{p,t})$,

2. The elements of $S$ are such that $S[1] \in [1,i_{p,1}]$, $S[a] \in [i_{p,a-1}+1,i_{p,a}]$ for $a \in [2,t]$, and $S[t+1] \in [i_{p,t}+1,d]$.

We prove Lemma 7.3 at the end of the section, and complete the proof of Theorem 1.7 using Lemma 7.3. Let $\text{IMM}_{w,d} = T_1 + \ldots + T_s$, where a term $T_p$ is as given by Lemma 7.3. For $j \in [d] \setminus S$, in each term $T_p$ $p \in [s]$, substitute the variables in $x_j$ such that $Q_j$ becomes a $w \times w$ identity matrix if $j \neq d$, and substitute the variables in $x_d$ such that $Q_d$ becomes a $w$ length unit vector with a one in the first entry if $d \in [d] \setminus S$. Let $\tilde{T}_p$ denote the polynomial $T_p$ after these substitutions. From properties 1 and 2 of the above lemma, $\tilde{T}_p$ is equal to the product of $t+1$ linear forms $\ell_{p,1} \cdots \ell_{p,t+1}$, where $\ell_{p,a}$ is a linear form in $x_{S[a]}$ variables for all $a \in [t+1]$. Further, let $f = \sum_{p \in [s]} \tilde{T}_p$. Then (up to renaming of variables) $f$ is equal to the iterated matrix multiplication polynomial of width $w$ and length $t+1$. Hence, $\sum_{p \in [s]} \tilde{T}_p$ is a multilinear depth three expression computing $\text{IMM}_{w,t+1}$ and since $t = \Omega\left(\frac{\log d}{\log \frac{d}{w}}\right)$, Theorem 1.4 implies $s = w^{\Omega(t+1)} = w^{\Omega\left(\frac{\log d}{\log \frac{d}{w}}\right)}$.

**Proof of Lemma 7.3**

To prove the lemma we introduce some notations. Let $e$ denote the root of $\varphi$ which we assume without loss of generality is a sum node. For a gate $v$ in $\varphi$, $\varphi_v$ denotes the polynomial computed at gate $v$. Let $v$ in $\varphi$ be a product gate with the associated interval $[i_1,i_2]$. Then there is an $i_3 \in [i_1,i_2-1]$ such that the children of $v$ are associated with intervals $[i_1,i_3]$ and $[i_3+1,i_2]$. The interval $[i_1,i_3]$ is called the left part of the interval $[i_1,i_2]$, and the child of $v$ associated with this interval is denoted as $v_l$. Similarly $[i_3+1,i_2]$ is called the right part of the interval $[i_1,i_2]$, and the child of $v$ associated with this interval is denoted as $v_r$. If the product
gate is denoted using an additional index like \( v_a \) then we denote the children of \( v_a \) as \( v_{a,\ell} \) and \( v_{a,r} \) respectively. A right-path \( p \) is a path \((e \rightarrow v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_l)\) in \( \varphi \) from the root node \( e \) to a leaf node \( v_l \) such that, if \( v_a \) is a product node for some \( a \in [1, l-1] \) then \( v_{a+1} = v_{a,r} \). We use \( P \) to denote the set of all right-paths in \( \varphi \). The proof of Observation 7.1 follows from the fact that corresponding to each leaf node in \( \varphi \) there is one path from the leaf node to \( e \).

**Observation 7.1** The cardinality of \( P \) is at most \( s \).

We will express the polynomial computed by \( \varphi \) as a sum of \(|P|\) terms, where each term corresponds to a right-path in \( P \). Let \( p \) be a right-path \( s \rightarrow v_1 \rightarrow \ldots \rightarrow v_l \) in \( P \), and let \( u_1, u_2, \ldots, u_k \) be the product gates in \( p \) ordered from left to right. Observe that \( k \leq l-1 \), as \( v_l \) is a leaf node labelled by a variable in \( x_d \). The term corresponding to \( p \), denoted as \( T_p \), is defined as \( \varphi_{u_1,\ell} \cdot \varphi_{u_2,\ell} \cdot \ldots \cdot \varphi_{u_k,\ell} \cdot \varphi_{v_l} \). The proof of Observation 7.2 follows from the definitions of \( P \) and the term corresponding to a right-path in \( P \).

**Observation 7.2** \( \text{IMM}_{w,d} = \sum_{p \in P} T_p \).

The proof of Lemma 7.3 is almost complete from Observations 7.1 and 7.2 and all that remains to show is that a term \( T_p, p \in P \) satisfies properties 1 and 2 in Lemma 7.3. In Observation 7.3 we use the \( \alpha \)-balance property of \( \varphi \) to prove that the number of product gates in a right-path in \( \varphi \) is at least \( t \).

**Observation 7.3** Let \( p \) be a right-path \( s \rightarrow v_1 \rightarrow \ldots \rightarrow v_l \) in \( P \), and let \( u_1, u_2, \ldots, u_k \) be the product gates in \( p \) ordered from left to right. Then \( k > t \).

**Proof:** Since \( u_k \) is the last product node in path \( p \), the interval associated with it is \([d, d]\). Further as \( \varphi \) is \( \alpha \)-balanced, the interval associated the \( t \)-th product gate in the path \( p \) has length at least \([\alpha t d] \geq 2 \). Since the length of the interval associated with \( u_k \) is equal to one, \( k > t \).

Observation 7.4 is proved by just using the interval property of \( \varphi \).

**Observation 7.4** Let \( T_p = \varphi_{u_1,\ell} \cdot \varphi_{u_2,\ell} \cdot \ldots \cdot \varphi_{u_k,\ell} \cdot \varphi_{v_l} \). Then there are indices \( 1 \leq j_{p,1} < j_{p,2} < \ldots < j_{p,k} = d-1 \) such that \( \varphi_{u_1,\ell} \cdot \ldots \cdot \varphi_{u_k,\ell} \cdot \varphi_{v_l} \) is a \( k+1 \)-set-multilinear product in \((j_{p,1}, \ldots, j_{p,k})\).

**Proof:** We prove this inductively starting with \( u_1 \). Since \( u_1 \) is the first product gate in path \( p \) from root to leaf, the interval associated with \( u_1 \) is \([1, d]\). Let \( j_{p,1} \in [d-1] \) be such that the intervals associated with \( u_{1,\ell} \) and \( u_{1,r} \) are \([1, j_{p,1}]\) and \([j_{p,1} + 1, d]\) respectively. Thus \( \varphi_{u_{1,\ell}} \) is a set-multilinear polynomial in the variable sets \( x_1, \ldots, x_{j_{p,1}} \). Assume there are
indices $j_{p,2} < \ldots < j_{p,k}$ where $b < k$, satisfying the following properties: $\varphi_{u_{1,\ell}}$ is a set-multilinear polynomial in the variable sets $x_1, \ldots, x_{j_{p,1}}$, $\varphi_{u_{2,\ell}}$ is a set-multilinear polynomial in the variable sets $x_{j_{p,1}+1}, \ldots, x_{j_{p,2}}$, $\ldots$, $\varphi_{u_{b,\ell}}$ is a set-multilinear polynomial in the variable sets $x_{j_{p,b-1}+1}, \ldots, x_{j_{p,b}}$, and $\varphi_{u_{b,r}}$ is a set-multilinear polynomial in the variable sets $x_{j_{p,b}+1}, \ldots, x_t$. Since $u_{b+1}$ is the next product node after $u_b$ in path $p$, the interval associated with $u_{b+1}$ is $[j_{p,b} + 1, d]$.

Let $j_{p,b+1} \in [d-1]$ be such that the intervals associated with $u_{b+1,\ell}$ and $u_{b+1,r}$ are $[j_{p,b} + 1, j_{p,b+1}]$ and $[j_{p,b+1} + 1, d]$ respectively. Hence, $\varphi_{u_{b+1,\ell}}$ is a set-multilinear polynomial in the variable sets $x_{j_{p,b}+1}, \ldots, x_{j_{p,b+1}}$. Finally, $j_{p,k} = d-1$ follows by noticing that $v_i$ is a leaf node and the interval associated with it is $[d, d]$.

Although the indices $\{j_{p,1}, j_{p,2}, \ldots, j_{p,k}\}$ in Observation 7.4 ensures that a term $T_p, p \in P$ satisfies property 1 in Lemma 7.3, it might not satisfy property 2. We will choose a set of indices $\{i_{p,1}, \ldots, i_{p,t}\}$ (from Observation 7.3 $k > t$) with respect to which the term $T_p$ satisfies both properties 1 and 2. First, observe that for any set $\{i_{p,1}, \ldots, i_{p,t}\} \subseteq \{j_{p,1}, \ldots, j_{p,k}\}$, where $1 \leq i_1 < \ldots < i_t < d-1$, we can regroup $T_p = \varphi_{u_{1,\ell}} \cdots \varphi_{u_{k,\ell}} \cdot \varphi_{v_1}$ by combining adjacent polynomials appropriately into $t+1$ set-multilinear polynomials $f_{p,1}, \ldots, f_{p,t+1}$ such that, $T_p = f_{p,1} \cdots f_{p,t+1}$ is a $(t+1)$-set-multilinear product in $(i_1, \ldots, i_t)$. We now show how to choose $i_{p,1}, \ldots, i_{p,t}$.

We use the $\alpha$-balance property to determine the set $i_{p,1}, \ldots, i_{p,t}$ with respect to which the term $T_p$ satisfies both properties 1 and 2. For $a \in [t]$, $i_{p,a} = \max_{b \in [t]} \{ j_{p,b} \leq \lfloor (1 - \alpha^a) d \rfloor \}$. Since $\varphi$ is $\alpha$-balanced, it is easy to observe that $j_{p,1} < d - \lceil \alpha d \rceil = \lfloor (1 - \alpha) d \rfloor$. Hence, to argue the correctness of this choice it is sufficient to show that $i_{p,1}, i_{p,2}, \ldots, i_{p,t}$ chosen as above satisfies $i_{p,1} < i_{p,2} < \ldots < i_{p,t}$. We prove this in Claim 7.4 and thereby conclude the proof of Lemma 7.3.

**Claim 7.4** Let $i_{p,a} = j_{p,b}$ for some $a \in [t-1]$ and $b \in [k-1]$. Then $j_{p,b} \leq \lfloor (1 - \alpha^a) d \rfloor$ and $j_{p,b+1} \leq \lfloor (1 - \alpha^{a+1}) d \rfloor$ and this in particular implies $i_a < i_{a+1}$.

**Proof:** Since $i_{p,a} = j_{p,b}$, it follows that $j_{p,b} \leq \lfloor (1 - \alpha^a) d \rfloor$. From the proof of Observation 7.4, we know that the interval associated with node $u_{b+1}$ in right-path $p \in P$, is $[j_{p,b} + 1, d]$ and the intervals associated with its children $u_{b+1,\ell}$ and $u_{b+1,r}$ are $[j_{p,b} + 1, j_{p,b+1}]$ and $[j_{p,b+1} + 1, d]$ respectively. The length of the interval associated with $u_{b+1}$ is $d - j_{p,b} \geq d - \lfloor (1 - \alpha^a) d \rfloor \geq \alpha^a d$, as $d \in \mathbb{N}$. Since $\varphi$ is $\alpha$-balanced the length of the interval associated with $u_{b+1,r}$ is at least $\lceil \alpha(d - j_{p,b}) \rceil$. Hence, $d - j_{p,b+1} \geq \lceil \alpha(d - j_{p,b}) \rceil \geq \lceil \alpha^{a+1} d \rceil$ implying $j_{p,b+1} \leq d - \lceil \alpha^{a+1} d \rceil \leq \lfloor (1 - \alpha^{a+1}) d \rfloor$. 

158
Chapter 8

Future Work

Circuit reconstruction: Our main contributions on problems related to learning in arithmetic circuit complexity are the algorithms for equivalence testing of the iterated matrix multiplication polynomial, average-case linear matrix factorization and average-case reconstruction of low-width algebraic branching programs. There are a few interesting questions that naturally arise from these works that need to be addressed in the future. We make a note of these questions below.

1. The algorithm in Theorem 1.3 solves average-case ABP reconstruction for \((w, d, n)\)-ABP \(X_1 \cdots X_d\), when \(w \leq \sqrt[2]{n}\). The running time of this algorithm is \((d^{w^3 n} \beta)^{O(1)}\), where \(\beta\) is the bit length of the coefficients of the input polynomial. There is one step in the algorithm that computes a bases of the space spanned by the linear form in \(X_1\) and \(X_d\) (denoted \(X_1\) and \(X_d\) respectively) which takes \((d^{w^3 n} \beta)^{O(1)}\) time and the remaining steps run in \((dn^\beta)^{O(1)}\) time. An efficient algorithm to compute the corner spaces \(X_1\) and \(X_d\) of a random \((w, d, n)\)-ABP will bring down the overall complexity of the algorithm. Since in Lemma 6.2 we show that with high probability these corner spaces are unique for a random \((w, d, n)\)-ABP when \(w \leq \sqrt[2]{n}\), there is hope that computing them efficiently is possible. An approach in this direction could have been the study of the structure of the Lie algebra of a polynomial computed by a random \((w, d, n)\)-ABP when \(w \leq \sqrt[2]{n}\), but it turns out that the Lie algebra of such a polynomial is trivial with high probability.

2. A polynomial computed by a \((w, d, n)\)-ABP \(X_1 \cdots X_d\) is zero modulo the linear space \(X_i\), for every \(i \in [d]\). Using this weakness of the ABP it is easy to show that the width of any homogeneous ABP computing \(x_1^n + \ldots + x_n^n\) is at least \(\sqrt[2]{n}\). Theorem 1.3 solves reconstruction for non-degenerate \((w, d, n)\)-ABPs \(X_1 \cdots X_d\), when \(w \leq \sqrt[2]{n}\), by turning this weakness into a non-degeneracy condition. This is achieved by ensuring that the
linear forms in a matrix $X_i$ are $\mathbb{F}$-linearly independent, that is $\dim(X_i) = w^2$ for $i \in [2, d-1]$ and $\dim(X_i) = w$ for $i \in \{1, d\}$. [Kum17] proved a $\frac{n}{2}$ lower bound on the width of any homogeneous ABP computing $x_1^n + \ldots + x_n^n$. The lower bound proof in [Kum17] exploits the fact that the first order derivatives of a polynomial computed by a homogeneous $(w, d, n)$-ABP is zero modulo the following $2w$ polynomials for any $i \in [d-1]$: $X_1 \cdots X_i(\ast, j)$ and $X_{i+1}(j, \ast) \cdots X_d$, $j \in [w]$, where $X_i(\ast, j)$ (respectively $X_{i+1}(j, \ast)$) denotes the $j$-th column (respectively row) of $X_i$ (respectively $X_{i+1}$). It remains open to design an algorithm for non-degenerate/average-case reconstruction for ABPs of width at most $\frac{n}{2}$ by defining a non-degeneracy condition based on the weakness of the ABP model exploited by [Kum17].

3. The proof Theorem 1.2 shows that a random $(w, d, n)$-matrix product is a pure product when $w \leq \sqrt{\frac{n}{2}}$, whose linear factorization is unique (in the sense mentioned in the third remark after Theorem 1.2). For $w$ significantly larger than $\sqrt{n}$, say $w = n^2$, can we show that linear factorization of a random $(w, d, n)$-matrix product is unique?

4. In the third remark after Theorem 1.1b, the polynomial IMM'$_{w,d}$ was defined as the trace of a product of $d, w \times w$ symbolic matrices. As noted in the remark, using the techniques from this work [Mur19] reduces the equivalence testing problem for IMM'$_{w,d}$ to multilinear equivalence equivalence testing for IMM'$_{w,d}$. Also using the determinant equivalence testing algorithm in [GGKS19], [Mur19] is able to solve the multilinear equivalence testing of IMM'$_{w,d}$ over $\mathbb{Q}$ and any finite field of large enough characteristic. Over $\mathbb{Q}$ the output matrices are over a degree $w$ extension of $\mathbb{Q}$, and over finite fields the output matrices are over the base field. An immediate future direction that follows from the work of [Mur19], is to give an efficient algorithm for the multilinear equivalence testing of IMM'$_{w,d}$ over $\mathbb{Q}$, where the output matrices are over $\mathbb{Q}$ instead of a degree $w$ extension of $\mathbb{Q}$.

**Lower bounds:** We prove a $w^{\Omega(d)}$ lower bound on multilinear depth three circuit computing IMM$_{w,d}$ in Theorem 1.4. Subsequent to our work, a $w^{\Omega(\sqrt{d})}$ lower bound on the size of multilinear depth four circuit computing IMM$_{w,d}$ was proved in [KST16b]. A $2^{\Omega(\Delta d^\frac{1}{4})}$ lower bound on the size of product-depth $\Delta$ multilinear formulas computing IMM$_{2,d}$ was proved in [CLS19]. Extending these results to a $w^{\Omega(\log d)}$ size lower bound on multilinear formulas computing IMM$_{w,d}$ will imply VBP $\neq$ VF, when $d = w^{\log w}$ and $d' = O\left(\frac{\log w}{\log \log w}\right)$. Even a super-polynomial size lower bound on multilinear formulas computing IMM$_{w,d}$ for $d' = O\left(\frac{\log w}{\log \log w}\right)$ will imply VBP $\neq$ VF.
In Theorem 1.7, we prove a $\Omega(\frac{\log d}{\log \frac{1}{\alpha}})$ size lower bound on $\alpha$-balanced interval set-multilinear formulas computing $\text{IMM}_{w,d}$. For $\alpha \geq d^{\frac{1}{\omega(1)}}$ and $\alpha < \frac{1}{2}$ this is a super-polynomial lower bound. A possible direction for the future is to prove a super-polynomial lower bound for much smaller values of $\alpha$. A super-polynomial size lower bound on interval set-multilinear formulas (without the $\alpha$-balance restriction) computing $\text{IMM}_{w,d}$ implies a separation between ABPs and homogeneous formulas in the non-commutative setting.

**Polynomial identity testing:** [dOSIV16] gave subexponential time blackbox PIT for multilinear depth three circuits. In our Masters thesis, we gave a quasi-polynomial time blackbox PIT for an interesting subclass of multilinear depth three circuits which are superposition of constantly many set-multilinear depth three circuits and simultaneously also a sum of constantly many set-multilinear depth three circuits (see Section 1.2.4). We could attempt to make progress on the quest for a quasi-polynomial time blackbox PIT for multilinear depth three circuits by trying to give a quasi-polynomial time blackbox PIT for multilinear depth three circuits that are superposition of constantly many set-multilinear depth three circuits.
Bibliography


BIBLIOGRAPHY


BIBLIOGRAPHY


164


[KNS16] Neeraj Kayal, Vineet Nair, and Chandan Saha. Separation between read-once oblivious algebraic branching programs (roabps) and multilinear depth three


BIBLIOGRAPHY


BIBLIOGRAPHY


[Shi16] Yaroslav Shitov. How hard is the tensor rank? arXiv, abs/1611.01559, 2016. 10


BIBLIOGRAPHY


BIBLIOGRAPHY

