Elimination Algorithms for Skew Polynomials with Applications in Cybersecurity

Raqeeb Rasheed

Department of Computer Science

A thesis submitted in partial fulfilment of the requirements of

Nottingham Trent University for the degree of

Doctor of Philosophy

January 2026

Abstract

This thesis details a comprehensive investigation of polynomial resultants, advancing algebraic techniques for non-commutative skew (Ore) polynomials and their novel applications in cryptography. It addresses critical needs in symbolic computation for effective non-commutative tools (such as resultant) and in cybersecurity for enhanced cryptographic primitives, including novel secret sharing schemes and key-exchange protocols.

A core component is the development of skew resultants for multivariate (initially bivariate) skew polynomials, extending existing univariate theories. Key algebraic contributions include new formulations, elimination methods, and establishing links between resultants and operator evaluation maps. Two primary computational methodologies are detailed; a direct matrix approach and a modular method using evaluation and interpolation. Applicability is also demonstrated for solving skew polynomial systems and within structures such as almost commutative rings.

Building on this algebraic foundation, the study introduces the application of polynomial resultants in cryptography. It presents a detailed development of two novel (t, n)-threshold secret sharing schemes and also proposes a Diffie-Hellman-

like key-exchange protocol that utilises the inherent commutativity of skew resultants. The two schemes offer different trade-offs; one uses numerical shares for efficiency, while the other employs symbolic polynomial shares for enhanced security and potential post-quantum advantages. The basis for these schemes is a specialised algorithm, also developed in this study, for constructing input polynomials with precise resultant properties. Both schemes incorporate built-in verification and adversary detection derived from these fundamental properties.

The theoretical correctness and practical feasibility of all developed methods are validated by thorough proofs and illustrative examples, including adversarial scenarios. This work significantly advances non-commutative resultant theory and its practical application, offering new tools for symbolic computation and cryptography while also suggesting avenues for future research into n-variate generalisations and further cryptographic development.

Acknowledgments

I would like to express my sincere gratitude to my supervisors, Dr. Ali Safaa Sadiq and Dr. Omprakash Kaiwartya, for their invaluable feedback and advice.

I am also grateful to the team at the Cyber Security Research Group (CSRG), Department of Computer Science, for their friendly collaboration, as well as to my family and friends for their support during this study. In particular, I thank my wife for her patience and encouragement; moreover, the presence of my children provided the hope and motivation I needed to continue. Additionally, this work is dedicated to my mother and the loving memory of my father who passed away several months before its completion, it is a profound regret that he could not be here to see the end of this journey.

Publications

The research study presented in this thesis has contributed to the following peer-reviewed papers and pre-prints:

Peer-reviewed Journal Papers

- Raqeeb Rasheed[1], Ali Safaa Sadiq, Omprakash Kaiwartya, "Elimination Algorithms for Skew Polynomials with Applications in Cybersecurity", *Mathematics*, 12(20), 3258.
  - The paper was nominated as a *Feature Paper* and selected as an *Editor's Choice* article by the MDPI journal *Mathematics*. Additionally the paper was selected to be on the front page of the journal website for banner promotion, including a direct hyperlink to the paper.
  - The study also received the runner-up prize at STAR 2024 Conference, the NTU Science and Technology Annual Research Conference.
- Raqeeb Rasheed[1], Ali Safaa Sadiq, Omprakash Kaiwartya, "Resultant-Based Threshold Secret Sharing Schemes with Verification and Adversary Detection", (Submitted).

Peer-reviewed Conference Papers

- Raqeeb Rasheed, "Resultant-based Elimination for Skew Polynomials", in *23rd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, Dec. 2021, pp. 11-18.

[1]Primary author.

Pre-print Papers

- Raqeeb Rasheed, "Resultant-based Elimination in Ore Algebra", *arXiv preprint arXiv:2105.14799*, 2022.

Contents

Notations

This section provides a list of common mathematical notations used throughout this study. Specific notations may be further clarified within the context of their introduction.

Symbol	**Description**
$\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$	The sets of natural, integer, rational, real, and complex numbers.
$\mathcal{F}$	A (skew) field, also referred to as a division ring.
$\mathcal{K}$	A field, often used when $\mathcal{F}$ is specifically a skew field.
$\mathcal{F}^\times$	The multiplicative group of non-zero elements of $\mathcal{F}$.
$[\mathcal{F}^\times, \mathcal{F}^\times]$	The commutator subgroup of $\mathcal{F}^\times$.
$\mathcal{R}$	A general ring, possibly non-commutative.
$\mathcal{R}[\theta]$	The ring of polynomials in the indeterminate θ over the ring $\mathcal{R}$.
$\mathcal{F}[\theta]$	The ring of polynomials in θ over the field $\mathcal{F}$.
$\mathcal{F}(\theta)$	The field of rational functions in θ over the field $\mathcal{F}$.
$\mathcal{F}[\theta_1, \ldots, \theta_n]$	The ring of polynomials in n indeterminates $\theta_1, \ldots, \theta_n$ over $\mathcal{F}$.
$\mathcal{F}(\theta_1, \ldots, \theta_n)$	The field of rational functions in n indeterminates over $\mathcal{F}$.
$\mathcal{F}[\theta; \sigma, \delta]$	The Ore polynomial ring over $\mathcal{F}$ in θ defined by the pair (σ, δ).
$\mathcal{F}(\theta; \sigma, \delta)$	The skew field of fractions of $\mathcal{F}[\theta; \sigma, \delta]$.
$\mathcal{R}[\theta_1; \sigma_1][\ldots][\theta_n; \sigma_n]$	Iterated Ore polynomial ring in n indeterminates.
$\mathcal{E}/\mathcal{F}$	A field extension $\mathcal{E} \supseteq \mathcal{F}$.
$M_n(\mathcal{R})$ or $\mathcal{R}^{n\times n}$	The ring of $n \times n$ matrices with entries in $\mathcal{R}$.
$\mathrm{GL}_n(\mathcal{R})$	The general linear group of invertible $n \times n$ matrices over $\mathcal{R}$.

$\mathrm{SL}_n(\mathcal{R})$	The special linear group of $n \times n$ matrices over $\mathcal{R}$ with determinant 1.
C_P	The companion matrix of a polynomial P.
A^T	The transpose of matrix A.
$\det(A)$	The standard determinant of matrix A (for commutative entries).
$\mathrm{Det}(A)$	The Dieudonné determinant of matrix A (for skew field entries).
$d_1 \mid d_2$	d_1 divides d_2.
$N \lhd G$	N is a normal subgroup of group G.
G/N	The quotient group of N in G.
$[a, b]$	The multiplicative commutator $aba^{-1}b^{-1}$.
$\mathrm{End}(\mathcal{R})$	The ring of endomorphisms of $\mathcal{R}$.
$\mathrm{Aut}(\mathcal{R})$	The group of automorphisms of $\mathcal{R}$.
$\bigoplus$	Direct sum.
$\prod$	Direct product.
$\mathrm{Cl}(a)$	The conjugacy class of an element a.
$V(f)$	The set of all (right) roots of a polynomial f.
$f(\theta_1, \theta_2)$	A generic polynomial in the non-commutative theory chapters.
$F(x, y)$	A specific polynomial in a commutative application (Secret Sharing).
$F(\theta_1, \theta_2)$	A specific polynomial in a non-commutative application (DH).
$\deg(f)$	Degree of polynomial f.
$\mathrm{lc}(f)$	Leading coefficient of polynomial f.
$\mathrm{rrem}(f, g)$	Right remainder of f divided by g.
$\mathrm{rquo}(f, g)$	Right quotient of f divided by g.
$\mathrm{gcrd}(f, g)$	Greatest common right divisor of f and g.
$\mathrm{lclm}(f, g)$	Least common left multiple of f and g.
$\mathrm{Res}_x(F, G)$	Resultant of F and G with respect to x.
σ	A ring automorphism.
δ	A σ-derivation.
θ	An indeterminate in an Ore polynomial ring.

Chapter 1

Introduction

The rapid evolution of digital technologies alongside with increasingly advanced adversarial capabilities, necessitates continual innovation in computational and cryptographic research. This study addresses this need by developing algebraic foundations that enhance both theoretical understanding and practical security protocols.

Polynomial computations are fundamental to modern computer algebra systems and have wide-ranging applications across mathematics, coding theory, control systems, and cryptography. In particular, this study focuses on a specialised class of non-commutative polynomials known as *Ore polynomials* (or skew polynomials) [111]. These generalise classical polynomial structures by introducing non-commutativity, making them particularly suitable for modelling operator systems such as differential and difference equations.

While the classical resultant is well established in the commutative setting [37], its extension to non-commutative multivariate systems remains challenging. This research aims to advance the algebraic understanding of resultants for skew polynomial systems and to harness these developments to construct secure cryptographic primitives. The approach establishes a conceptual link between abstract algebraic elimination theory and practical cryptographic design.

Building on these novel algebraic foundations, the study introduces the application of polynomial resultants to address ongoing challenges in cryptography

by developing two new primitives.

First, motivated by vulnerabilities in existing protocols [30, 46, 70, 78], this research introduces the first (t, n)-threshold secret sharing schemes constructed from bivariate polynomial resultants. The two proposed schemes, *Dealer-Side Scaling* and *Participant-Side Scaling*, incorporate built-in verification and adversary detection, with the latter also offering potential post-quantum advantages. This work extends the foundational concepts of Shamir and Blakley [8, 120] to a new algebraic domain, with wide-ranging relevance to applications like secure data storage and multiparty computation [7, 41, 69, 94, 98, 133].

Second, a novel *Diffie-Hellman-like* key-exchange protocol is proposed. It utilises inherent properties of skew resultants to enable a simple *conjugation-style* key-exchange. This design avoids the need to construct special subsets of commuting polynomials, a notable challenge and potential vulnerability in earlier non-commutative proposals [18].

Overall, this study traverses from foundational explorations in non-commutative algebra, focusing on the development of comprehensive resultant theories, to the practical application of resultant concepts in addressing challenges in cryptography. This research aims to contribute new theoretical insights, efficient computational methods, and novel security applications, highlighting the versatility and power of polynomial resultants. The focus on bivariate cases is strategically chosen, not only for detailed treatment but also for its core role in potential recursive techniques applicable to more general multivariate systems.

This thesis therefore presents several major contributions, with additional technical results detailed in their respective chapters. The following provides a high-level overview of the main achievements in both non-commutative algebra and cryptography:

- A new theoretical and algorithmic framework for bivariate skew resultants is developed.

- Efficient implementation strategies are proposed to overcome the problem

of intermediate expression swell common in symbolic computation.

- Novel resultant-based secret sharing schemes are constructed, featuring built-in mechanisms for share verification and adversary identification.
- A Diffie-Hellman-like key-exchange protocol is designed that utilises inherent properties of skew resultants to avoid the need for pre-computed commuting sets.

The structure of this thesis will guide the reader through these interconnected research areas. Following this introduction, Chapter 2 provides a review of relevant literature. Chapter 3 details mathematical preliminaries and background on Ore polynomials and resultant theory. Subsequent chapters present the development of resultant-based elimination methods for skew polynomials, including the specific challenges, novel formulations, and algorithmic contributions (Chapter 4), followed by methods for efficient computation of these resultants using evaluation and interpolation (Chapter 5). This is followed by Chapter 6 which provides an experimental analysis and performance benchmarks for the proposed methods. The study then presents the detailed design of new resultant-based secret sharing schemes, their underlying polynomial construction algorithms, security analyses, and comparisons (Chapter 7). Following this, Chapter 8 provides a novel Diffie–Hellman-like key-exchange protocol. Finally, Chapter 9 concludes the thesis by summarising the key findings and contributions of the entire work and outlining promising directions for future research.

1.1 Motivation of the Research

The motivation for this research stems from the significant role of skew (Ore) polynomials in various scientific and mathematical disciplines, the persistent challenges associated with efficiently solving systems of such non-commutative polynomials, and the growing necessity for robust data security in an era of advancing digital technologies. This investigation is driven by the need for more

advanced computational tools and novel cryptographic primitives. Several key factors underpin this study:

- **Advancing Non-Commutative Resultant Theory and Algorithms:** Current resultant methods for Ore polynomials are largely limited to the univariate case [52, 75, 79, 91, 93]. There is a clear need to extend these to effective elimination techniques for *multivariate* skew polynomial systems, which are crucial for tackling more complex problems. Existing commutative multivariate methods [51, 80, 123] do not directly translate to the non-commutative Ore framework. This necessitates the development of efficient algorithms specifically optimised for multivariate skew systems (particularly in the difference case, $\delta = 0$), adapting existing principles and forging new connections, such as the proposed link between skew resultants and operator evaluation maps.

- **Developing Modular Methods and Leveraging Symbolic-Numeric Techniques:** Modular techniques are highly effective in commutative algebra [5, 24, 38, 112] but have seen limited application in Ore algebra, primarily targeting coefficients over commutative base rings [28, 29, 42, 92]. This gap restricts the development of efficient algorithms. This research is motivated by the opportunity to revisit classical methods like resultant computations by incorporating recent advancements in non-commutative evaluation and interpolation strategies [95, 101] to handle the indeterminates themselves more effectively.

- **Extending Foundational Work and Tackling Algebraic Challenges:** Building upon earlier studies [112], a key motivation is to provide a comprehensive understanding of the resultant-based elimination process for skew polynomials, including the thorough development of the often-overlooked interpolation stage. This involves directly addressing the inherent challenges of non-commutative algebra, such as developing effective analogues

for essential tools like multivariate resultants and navigating the complexities of non-standard evaluation maps and determinant formulations.

- **Validating Computational Strategies:** Practical comparative analysis of computational strategies is essential. While direct symbolic methods based on triangularising a Sylvester matrix are theoretically straightforward, they suffer from severe intermediate expression swell. This study is motivated to implement and benchmark both this direct approach and a modular evaluation-interpolation technique, aiming to experimentally validate the greater scalability of the latter and demonstrate its viability for large problem instances despite theoretical complexities.

- **Advancing Cryptographic Applications and Addressing Vulnerabilities:** The inherent computational complexity of skew polynomial systems offers promising avenues for cryptography [18], especially given the scarcity of attack methodologies for non-commutative systems. This research aims to establish novel resultant-based techniques for applications like secret sharing and key exchange. A critical driver is the need to address security vulnerabilities in existing threshold secret sharing schemes [30, 46, 78] by developing schemes inherently resistant to attacks, such as those involving adversary impersonation.

- **Developing Enhanced, Efficient, and Post-Quantum Secure Schemes:** Beyond basic functionality, there is a motivation to develop schemes with built-in features like reliable share verification and adversary identification, utilising resultant properties such as scaling and evaluation. Additionally, improving participant efficiency (e.g., Dealer-Side Scaling) and exploring potential post-quantum security advantages through symbolic polynomial shares (e.g., Participant-Side Scaling) are key objectives.

- **Establishing Foundations and Demonstrating Feasibility:** The study is driven by the need to provide formal theoretical foundations, in-

cluding explicit resultant formulae and proofs of properties like the annihilation of common solutions. Furthermore, demonstrating practical feasibility through supporting algorithms (e.g., for polynomial construction) and broader applications (e.g., combinatorial identities) is a significant motivation. The focus on bivariate systems serves as a foundational step for recursive techniques applicable to general $n \times n$ systems, opening avenues for future research in optimisation and generalisation.

At its core, this research aims to provide a complete, theoretically solid, and practically applicable framework that utilises polynomial resultants, contributing valuable tools to both algebraic computation and cryptographic security.

1.2 Research Gaps and Questions

A thorough review of the existing literature concerning elimination theory for skew (Ore) polynomials and the application of algebraic techniques in cryptography, particularly secret sharing, reveals several important avenues for further investigation and development. These identified research gaps directly motivate the central questions addressed by the comprehensive research presented in this study:

1. **Gap: Underdevelopment of Multivariate Non-Commutative Resultant Theory.** The majority of existing resultant theories and computations for Ore polynomials are limited to the *univariate* case [52, 75, 79, 91, 93]. While commutative multivariate resultants are well-studied [51, 80, 112, 123], a comprehensive theoretical and algorithmic framework for their counterparts in *multivariate* skew polynomial systems remains less developed. This gap includes the need for a complete treatment of all algorithmic stages, such as an interpolation component, which has not been fully addressed in prior foundational work.

- **Research Question 1.1:** How can a novel, complete, and theoretically sound resultant-based elimination framework be developed for multivariate (initially bivariate) skew polynomial systems, ensuring that all algorithmic stages (including an efficient evaluation and interpolation component) are thoroughly developed and validated?
- **Research Question 1.2:** What are effective methods for defining and computing such multivariate skew polynomial resultants, for instance, by employing specialised determinants such as the Dieudonné determinant, or by adapting/combining existing univariate skew polynomial resultant concepts?
- **Research Question 1.3:** Can formal resultant formulae be derived for bivariate skew operator polynomials, and can theorems be established (with proof) confirming their fundamental properties, such as the annihilation of common solutions?

2. **Gap: Limited Application of Advanced Computational Techniques in Ore Algebra.** Modular methods, highly effective in commutative algebra for managing complexity [5, 24, 38, 112], have seen limited application in Ore algebra. Existing uses [28, 29, 42, 92] primarily focus on coefficients over commutative rings and do not typically incorporate *non-commutative evaluation and interpolation* techniques for the *indeterminates*. Furthermore, while non-commutative evaluation/interpolation is an emerging field [95, 101], its structure with resultant theory for multivariate skew systems is not yet fully explored.

   - **Research Question 2.1:** How can modular approaches be effectively integrated with non-commutative evaluation and interpolation methods to develop efficient algorithms for computing resultants of bivariate skew polynomials?
   - **Research Question 2.2:** What is the exact relationship between such

resultants and operator evaluation maps, and how can this relationship be used to create novel computational techniques, particularly within the difference case (when $\delta = 0$)?

- **Research Question 2.3:** What are the comparative effectiveness of different computational methodologies for these resultants, such as a direct Sylvester-style matrix approach versus a modular technique (using evaluation and interpolation)?

3. **Gap: Lack of Practical Implementation and Experimentally Performance Analysis.** While theoretical frameworks for multivariate skew resultants can be proposed, a notable gap exists in their practical implementation and comparative performance analysis within computer algebra systems. The scalability, practical limitations, and computational behaviour of different techniques (such as direct symbolic triangularisation versus modular evaluation-interpolation) are not established for this non-commutative context. Phenomena like intermediate expression swell in symbolic methods require practical investigation to validate the feasibility of proposed algorithms.

    - **Research Question 3.1:** What are the practical challenges and feasibility of implementing both a direct symbolic method and an evaluation and interpolation method for computing bivariate skew polynomial resultants in a computer algebra system (such as Maple)?
    - **Research Question 3.2:** How do the computational performance and scalability of the direct symbolic approach and the evaluation-interpolation approach compare experimentally across a range of input polynomials with varying degrees and coefficient complexities?
    - **Research Question 3.3:** What are the underlying causes of observed performance characteristics, such as the non-monotonic behaviour in symbolic methods, and how do they relate to the algebraic properties

of the intermediate computations?

4. **Gap: Addressing Inherent Complexities of Non-Commutative Algebra.** The inherent non-commutative nature of skew polynomials presents unique challenges, such as the non-standard behaviour of evaluation maps (which may not preserve multiplication) and difficulties in defining unique, computationally useful determinants. The application of resultant theory in specific contexts such as in almost commutative rings within the Ore algebra framework also requires further clarification.

   - **Research Question 4.1:** How can the challenges unique to non-commutative algebra, including the properties of evaluation maps and determinant computations, be effectively addressed in the development and application of resultant methods for bivariate skew polynomials?
   - **Research Question 4.2:** How does the concept of the resultant appear in almost commutative rings, and under what conditions can it be shown to be well-defined and computationally feasible for skew polynomials in this context?

5. **Gap: Security Vulnerabilities and a Lack of Fundamentally Diverse Methodologies in Secret Sharing Schemes.** Established threshold secret sharing schemes, despite widespread use, suffer from documented security vulnerabilities and limitations, particularly concerning adversarial actions during reconstruction or when participant numbers exceed the minimum threshold [30, 46, 70, 78]. Furthermore, the application of polynomial resultants to the construction of secret sharing schemes represents a largely unexplored research direction, potentially offering new security features and fundamental design flexibility.

   - **Research Question 5.1:** How can polynomial resultants be effectively applied to construct new (t, n)-threshold secret sharing schemes

with inherently stronger security properties, including built-in verification process and adversary detection capabilities?

- **Research Question 5.2:** Is it feasible to use fundamental properties of polynomial resultants (such as evaluation and scaling) to design and construct practical, accurate, and secure secret sharing schemes, and what are their comparative advantages and trade-offs (e.g., numerical shares for efficiency versus symbolic shares for potential post-quantum advantages)?
- **Research Question 5.3:** What algorithmic procedures are required to generate suitable pairs of bivariate Ore polynomials whose skew resultants provide sufficient cryptographic strength and have provably predictable degrees suitable for cryptographic schemes?

6. **Gap: Absence of a Simplified Diffie-Hellman-Like Protocol that Utilises Inherent Commutativity in Ore Algebras.** Existing non-commutative key-exchange schemes, such as the Burger-Heinle primitive [18] and the approach of Boucher et al. [11], present trade-offs. The former relies on a computationally expensive, pre-computed commuting subset, which incurs considerable offline cost, while the latter depends on solving difficult Key-Factorisation problems that can be complex to parameterise securely. A gap exists for a protocol that utilises a more natural source of commutativity within the Ore algebra itself, potentially eliminating both the pre-computation and parameterisation complexities.

   - **Research Question 6.1:** Can a Diffie-Hellman-like key-exchange protocol be designed whose secret keys are skew resultants that inherently commute, thus removing the need for any pre-computed commuting subsets or complex Key-Factorisation conditions?
   - **Research Question 6.2:** How can the security of such a resultant-based protocol be formally analysed and potentially reduced to a well-

defined hard algebraic problem, such as a constrained multivariate Ore-polynomial factorisation?

- **Research Question 6.3:** What are the potential performance characteristics of a resultant-based scheme compared with existing methods, and which implementation techniques could control expression swell in practical parameter ranges within multivariate Ore algebra?

7. **Gap: Foundational Tools and Formal Validation for Novel Cryptographic Schemes.** The practical effects of new cryptographic schemes, such as those based on resultants, requires specific foundational tools (e.g., algorithms for constructing suitable input polynomials with resultant properties) which are currently not available. Moreover, any novel cryptographic scheme necessitates exact validation of its correctness and security claims.

   - **Research Question 7.1:** What algorithmic procedures are necessary for a dealer to construct appropriate bivariate polynomials such that their resultant has a predefined degree and correctly encodes the secret for a (t, n)-threshold scheme?
   - **Research Question 7.2:** Can the correctness of such polynomial construction algorithms and the proposed resultant-based secret sharing schemes (including their verifiability and adversary identification capabilities) be formally proven and practically demonstrated through illustrative examples, including adversarial scenarios?

8. **Gap: Demonstrating Broader Applicability of Developed Algebraic Tools.** The practical feasibility and range of applications for newly developed resultant techniques for skew polynomials, beyond their foundational role in elimination theory or a single application area, require further demonstration.

   - **Research Question 8.1:** How can the feasibility and practical value of the derived bivariate skew polynomial resultant be demonstrated

in diverse applications, such as identifying new combinatorial identities or establishing cryptographic protocols based on multivariate skew resultant?

Addressing these research gaps and questions forms the core of the investigations presented in this thesis. The purpose is to provide comprehensive theoretical frameworks, reliable algorithmic solutions, and demonstrated applications, thereby contributing significantly to symbolic computation for non-commutative systems and to the design of novel, secure cryptographic techniques.

1.3 Research Aims and Objectives

In response to the identified need for enhanced security and novel approaches in secret sharing, and utilising the potential of polynomial resultants, this research has the following main aims and specific objectives:

Aims

The main aims of this research are:

1. To investigate and establish the novel application of polynomial resultants as a foundational mechanism for constructing secure and verifiable secret sharing schemes.

2. To develop new (t, n)-threshold secret sharing schemes based on this resultant framework that offer enhanced security features, including verification and adversary detection.

3. To investigate the feasibility of applying resultant-based techniques to the development of a novel Diffie-Hellman-like key-exchange scheme.

4. To demonstrate the practical feasibility and distinct advantages of these novel resultant-based cryptographic schemes through examples and analysis.

5. To develop a complete theoretical and computational framework for multivariate skew (Ore) polynomial resultants, encompassing their algebraic foundations, the integration of modular evaluation-interpolation techniques, and an experimental analysis of their practical performance.

Objectives

To achieve the above-mentioned aims, this study pursues two interrelated sets of objectives, grouped according to their primary cryptographic and algebraic–computational focus as following:

Cryptographic Objectives

1. To identify and formalise those properties of polynomial resultants (commutative and non-commutative) that are most suitable for the construction of cryptographic primitives, in particular threshold secret sharing and key-exchange mechanisms.

2. To design, develop, and present distinct resultant-based (t, n)-threshold secret sharing schemes (Dealer-Side Scaling and Participant-Side Scaling) with differing operational characteristics, including efficiency, verification overhead, and potential post-quantum security.

3. To construct explicit algorithms for generating bivariate polynomials whose resultants exhibit the required degree and constant-term properties for secure secret encoding and reliable scheme operation.

4. To establish formal security and correctness guarantees for the proposed schemes, including proofs of privacy, correctness of reconstruction, verifiability, and adversary identification based on algebraic checks derived from resultant properties.

5. To illustrate the practical operation of these schemes through worked examples and adversarial scenarios, and to compare their behaviour with existing

approaches where appropriate.

Algebraic and Computational Objectives

1. To develop a resultant-based elimination framework for bivariate skew polynomials, drawing on tools such as the Dieudonné determinant and the structure of almost commutative rings, and to determine conditions under which such resultants are well-defined.

2. To formulate constructive methods for computing the resultant of two bivariate skew polynomials over (skew) fields, including both direct Sylvester-matrix-based approaches and alternative strategies inspired by commutative elimination theory.

3. To establish and analyse new theoretical relationships between bivariate skew resultants and operator evaluation maps, and to exploit these relationships via non-commutative evaluation–interpolation techniques.

4. To design and validate a computationally efficient evaluation–interpolation methodology for bivariate skew polynomial resultants as a scalable alternative to direct symbolic triangularisation, including an assessment of its complexity and behaviour with respect to intermediate expression swell.

5. To implement the proposed algorithms in a computer algebra environment, provide illustrative examples (including applications beyond cryptography, such as combinatorial identities or simplification of operator systems), and carry out comparative evaluations of the different computational strategies where relevant.

Successfully achieving these objectives will yield new insights and practical tools in two principal areas: the design of secure, flexible, and efficient threshold secret sharing schemes, and the advancement of symbolic computation in non-commutative algebra. These contributions are intended to support substantial progress in both cryptography and algebraic computation.

1.4 Research Challenges

The development of novel secret sharing schemes based on polynomial resultants and the advancement of resultant theory for multivariate skew polynomials require addressing a number of distinct challenges:

1. **Foundational Framework:** A fundamental challenge lies in formulating and applying polynomial resultant theory to the distinct requirements of secret sharing and key exchange in a novel way. This includes adapting concepts like the Dieudonné determinant, and navigating the complexities where commutative analogues fail.

2. **Generalising Univariate Non-Commutative Resultant Theory:** A substantial challenge is the extension of mostly univariate Ore/skew polynomial resultant theories to the multivariate (initially bivariate) domain. This task is complicated by the following fundamental algebraic obstacles absent in the commutative setting:

    - **Non-commutativity and left/right structure:** In the commutative case there is a single, symmetric notion of a *common root* and of a common factor. For skew/Ore polynomials one must distinguish between left and right roots, left and right divisors, and left/right ideals. Even formulating f and g have a common factor in the *eliminated variable* is more delicate, and the algebraic conditions live in non-two-sided ideals. This already complicates any attempt to mimic classical elimination via ideals and projections.

    - **Non-Standard Evaluation:** Standard scalar substitution does not preserve multiplication in non-commutative rings, necessitating the formulation of complex operator evaluation maps to serve as valid homomorphisms.

    - **Determinant Ambiguity:** The classical determinant is ill-defined for non-commutative matrices; this requires adapting the Dieudonné

determinant, which is defined only modulo commutators, to extract a meaningful polynomial resultant.

- **Absence of Unique Factorisation:** Unlike their commutative counterparts, multivariate skew polynomial rings are generally not Unique Factorisation Domains (UFDs). This structural limitation prevents the application of standard resultant constructions that rely on factorisation properties (such as content-primitive part decomposition or Gauss's Lemma) to manage the complexity of multivariate systems.

Navigating these complexities to formulate novel combinations and algorithms represents a primary theoretical hurdle of this research.

3. **Security and Verification:** Designing resultant-based techniques that are inherently resistant to known attacks (e.g., exploiting multiparty reconstruction weaknesses) is critical. Integrating effective verification and adversary detection directly within the resultant-based framework, utilizing properties like evaluation and scaling, presents a significant challenge.

4. **Algorithmic Complexity and Implementation:** Effectively integrating modular arithmetic with non-commutative evaluation and interpolation is complex. Furthermore, implementing these algorithms for symbolic computation involves translating abstract theory into efficient code, building high-level procedures from algebraic primitives, and managing intermediate expression swell. Cross-verifying distinct methods (direct vs. modular) is a primary practical challenge.

5. **Scheme Construction and Diversity:** Developing reliable algorithms to generate specific input polynomials with exact resultant properties is a prerequisite. Crafting multiple schemes that utilise the same core concept but cater to different priorities (e.g., efficiency vs. post-quantum potential via numerical vs. symbolic shares) is a complex design task.

6. **Formal Validation:** Formally proving the correctness of the algebraic methods and cryptographic schemes, along with their security properties (privacy, verifiability, adversary detection), is fundamental to establishing their validity.

Successfully navigating these challenges is essential to realising the potential of polynomial resultants in advancing both non-commutative algebra and the field of cryptography.

Chapter 2

Literature Review

This chapter reviews the two primary domains that form the foundation of this research. The first section surveys the algebraic literature on elimination theory, Ore polynomials, and associated computational methods, emphasising both their strengths and their limitations for non-commutative elimination. The second section examines the cryptographic context, assessing established techniques, applications, and known vulnerabilities in secret sharing and non-commutative key-exchange protocols, thus motivating the novel resultant-based schemes developed in this work. Finally, specific attention is devoted to review existing Diffie-Hellman-like protocols over skew polynomial rings, thereby setting the stage for the novel key-exchange mechanism proposed later in this study.

2.1 Elimination Theory and Ore Polynomials

The algebraic foundation of this study is closely related to the concept of *characteristic sets*, which provide a classical technique for representing the solution set of a general polynomial system (denoted by V) as the union of zero sets of polynomial equations in triangular form. This triangular representation belongs to the broader framework of elimination theory, which plays a central role in both differential and difference algebra and forms the basis of many modern algorithms in computational algebraic geometry.

2.1.1 Classical Elimination Frameworks

The study of elimination theory developed from the necessity to process and solve systems of differential equations in a uniform manner. Differential algebra, introduced by Ritt [117], intended to treat these equations analogously to algebraic geometry through the use of differential ideals. Ritt's framework, while conceptually powerful, was more qualitative and not directed towards large-scale computation, as complexity bounds were rarely made explicit and coefficient growth was only indirectly controlled. Later developments in differential elimination have often retained this theoretical focus, which means that their direct applicability to high-dimensional, non-commutative operator systems remains limited. One of the aims of the present study is to retain the structural insights of Ritt's approach while focusing on elimination constructions (via Sylvester-like matrices and resultants) that lend themselves to concrete computation in non-commutative Ore settings.

Elimination methods have been extensively studied and consist of several approaches, including *Chow forms*, *characteristic sets*, and *resultant-based approaches*. Each of these brings particular advantages, but also exhibits limitations when one moves beyond the classical commutative framework, especially towards skew (Ore) polynomials and algorithms designed to cryptographic applications.

The Chow form, sometimes termed the Cayley or Cayley–Chow form [43], is an essential object in the elimination theory of algebraic geometry. It allows the encoding of projective varieties and has proved to be a fundamental tool in areas such as transcendental number theory [108]. Gelfand et al. [62] investigated Chow forms for sparse polynomials and highlighted their close connection with resultants and discriminants. However, Chow-form based elimination is primarily designed for commutative projective varieties and can be computationally expensive; even in the sparse setting, the size of the Chow form grows rapidly with the dimension and degree of the variety, and there is no straightforward extension to non-commutative polynomial rings. In particular, the matrix representations

underlying Chow forms are not directly compatible with Ore multiplication. Consequently, while the conceptual role of Chow forms informs this work at a high level, the present study focuses instead on resultant constructions that admit concrete Sylvester-type matrices and can be adapted to skew polynomial rings.

On the other hand, characteristic sets, functioning as an extended Gaussian elimination method, remained relatively unexplored for a considerable period until Wu's work on zero decomposition for polynomial equations appeared in [129]. Wu extended these methods to the automation of theorem proving [127], differential algebraic geometry [130], and a significant advancement was his development of a *factorisation-free* algorithm, which removed the dependence of Ritt's decomposition method on polynomial factorisation. This innovation reduced the problem of solving polynomial equations to solving a succession of univariate equations and led to a broad range of decomposition techniques for various classes of polynomial systems, including differential systems [12, 66, 77], difference systems [36, 58, 59, 60, 82], mixed differential–difference systems [57], and, more recently, regular differential chains [13].

Despite their generality, characteristic set methods have several limitations from the perspective of the present study. Their worst-case complexity grows rapidly with the number of variables and the degrees of the polynomials, and coefficient swell can be severe. Moreover, the theory and algorithms are primarily formulated for commutative settings, and they do not transfer directly to Ore algebras, where non-commutativity fundamentally changes the structure of leading terms and reduction. The extension of such decomposition techniques to non-commutative algebras (like Ore extensions) faces significant challenges due to the lack of unique factorisation and commutative properties, a gap this study aims to address via resultants. In this thesis, characteristic sets and the Ritt–Wu framework are therefore used mainly as a conceptual reference point for elimination, while the core technical contribution is to develop resultant-based and matrix-oriented elimination procedures specifically adapted to skew polynomials

and amenable to modular and parallel computation.

Resultant theory provides another major avenue for elimination. While univariate resultants eliminate one variable at a time from two polynomials, multivariate resultants can eliminate several variables simultaneously from multiple polynomials, serving as an algebraic criterion for the solutions. Many efficient algorithms for solving polynomial equations are based on resultants [16, 20, 51, 80]. These works, however, largely focus on commutative polynomial rings and aim for generality, often leading to high-degree constructions and large matrices whose size makes them challenging to deploy in specialised settings such as cryptographic protocol design. The present study takes a more targeted approach; it concentrates on bivariate (skew) resultants, for which one can exploit structural properties such as degree constraints and constant-term control that are particularly meaningful for secret encoding and verification.

Ore extended polynomial algebra to differential and difference contexts by introducing non-commutative *Ore polynomial rings* [109], which model linear differential and shift operators within a unified algebraic framework. His work has subsequently influenced a range of studies in differential algebra and operator theory, e.g. [6, 31, 55, 132]. Differential resultants, initially developed by Ritt for univariate differential polynomials [115] and generalised to multivariate cases [21, 118, 134], provide powerful theoretical tools for studying the solvability of differential systems. Nonetheless, many of these constructions are primarily theoretical (rather than constructive), and explicit algorithms are often restricted to relatively small systems or specialised forms (e.g., linear, homogeneous). Furthermore, their adaptation to non-commutative operator rings, where multiplication is skewed by an automorphism or derivation, has been limited. By contrast, the present work develops explicit algorithms for bivariate skew resultants in Ore settings, with a particular emphasis on constructible Sylvester-like matrices and Dieudonné determinants that can be implemented and analysed in a computational framework, which are essential for constructing more complex

cryptographic schemes.

2.1.2 Towards Non-Commutative Ore Settings

The problem statement motivating the algebraic investigations in this study can be viewed as an extension of the ideas of Rasheed [112, 113, 114] and Li [92] from commutative and univariate Ore settings to non-commutative, multivariate skew polynomial rings. Classical approaches such as Wu's characteristic set method [127, 128], based on Ritt's concept of ascending chains [115, 116], underscore the centrality of elimination for representing solution sets of polynomial systems.

However, while these frameworks establish theoretical validity, the practical application of elimination theory in this non-commutative setting presents distinct computational challenges. Classical methods, such as Gröbner bases or the aforementioned characteristic sets, often suffer from severe intermediate expression swell and exponential complexity when extended to non-commutative algebras. The iterative nature of these rewriting systems can lead to coefficient growth that renders computations intractable for systems of even moderate size. This computational bottleneck necessitates a shift towards a linear-algebraic framework. By reformulating the elimination problem in terms of resultants, one can leverage determinantal computations (e.g., the Dieudonné determinant) which are more amenable to modular arithmetic and evaluation-interpolation techniques.

Consequently, the present work departs from iterative rewriting methods by developing explicit resultant-based elimination algorithms for bivariate skew polynomials. This strategic choice allows for the control of bit-complexity and provides a pathway to efficient algorithms that bypass the limitations of standard non-commutative rewriting, with a specific view towards applicability in cryptographic constructions.

2.1.3 Modular and Evaluation-Interpolation Approaches

Modular techniques are frequently employed in computer algebra to control coefficient growth and reduce the cost of computations by working modulo primes or modulo regular chains. Combining modular approaches with elimination theory can significantly improve performance in practice. For example, in the commutative setting, this idea has been utilised by Rasheed [112], where resultants and GCDs were used to solve bivariate and trivariate polynomial systems, combining modular reduction with reconstruction. Xin et al. [90] employed polynomial GCDs modulo regular chains to obtain high-performance solvers, while Berberich et al. [5] proposed a matrix-based elimination method as an alternative to Collins's polynomial remainder sequences (PRS) for computing real solutions. These modular approaches demonstrate that hybrid symbolic–numeric and matrix-based methods can be highly effective; however, they remain confined to commutative polynomial rings and do not address the structural complications of Ore algebras.

In particular, modular arithmetic has not transferred effectively to Ore settings, primarily because of the algebraic incompatibility between reduction modulo p and the non-commutative skew structure. Unlike in commutative rings, the reduction map is not generally a ring homomorphism for skew polynomials, since the associated automorphisms and derivations (σ, δ) may become undefined or degenerate over finite fields.

A second obstruction arises in the reconstruction phase. Ore rings are not unique factorisation domains, so a given skew polynomial can admit many inequivalent factorisations. This undermines the canonical lifting behaviour exploited in the commutative case via the Chinese Remainder Theorem and Hensel lifting; consequently, recombining modular images becomes highly ambiguous unless very strong additional constraints are imposed on the operators.

Moreover, even in the commutative case, the complexity of reconstruction from modular images and the sensitivity to unlucky primes can limit robustness. One of the objectives of this thesis is therefore to adapt the spirit of these modular

and matrix-based methods to the non-commutative setting of skew polynomials, where *evaluation–interpolation* techniques and *Dieudonné-determinant*-based resultants can be combined to obtain more scalable elimination algorithms with controlled coefficient growth.

While modular methods are well studied in commutative and differential algebras, their systematic application in Ore algebras has received less attention. Notable contributions include Li's work [91, 92, 93] on subresultant theory and modular algorithms for univariate Ore polynomials and the modular computation of matrices of Ore polynomials by Cheng and Labahn [28, 29]. More recently, Decker et al. [42] developed a modular technique within a probabilistic algorithm for computing Gröbner bases in certain G-algebras. These results establish proof-of-concept frameworks for modular non-commutative computation, but they typically concentrate on GCRD computations or Gröbner bases in fairly general Ore-type structures, they focus on modular arithmetic for coefficients, rather than applying modular techniques to the non-commutative indeterminates themselves. The recent probabilistic algorithm for G-algebras in [42] suggests progress, yet the lack of robust non-commutative evaluation and interpolation methods has hindered broader application. They do not address the specific problem of bivariate skew polynomial elimination via resultants. Nor do they exploit the explicit degree and evaluation structure required for encoding secrets or keys in a cryptographic context. The present work builds on these foundational ideas but for the bivariate skew setting, where the resultant is made explicit via Sylvester-like matrices and evaluated through non-commutative interpolation.

Recent research on non-commutative *evaluation and interpolation* techniques, such as [95, 100, 101, 102], has provided powerful tools for coding theory over skew and linearised polynomial rings. These works clarify how to evaluate and interpolate skew polynomials at operator-like points and to construct codes with desirable distance properties. However, their primary focus is on coding applications and not on elimination or the construction of resultants. In particular,

the interaction between skew evaluation and structured Sylvester matrices for elimination remains underexplored. This study draws on the concept of evaluation–interpolation techniques to design algorithms that compute skew resultants by evaluating at suitable operator points and reconstructing the resultant via interpolation, thereby establishing a link between (skew) resultants and cryptography. Parallel computing is an increasingly important framework in high-performance computing, and there have been notable efforts to parallelise bivariate polynomial solvers in commutative algebra. Emeliyanenko's work [49, 50] demonstrated how GPUs can be harnessed to compute resultants for real algebraic geometry problems, while Moreno Maza and Pan [103] developed GPU-accelerated algorithms for polynomial systems over finite fields. These contributions show that resultant-based elimination is well suited to data-parallel architectures. Nevertheless, current parallel solvers primarily target commutative polynomials and do not address multivariate Ore polynomials or skew resultants. In particular, the design of GPU-amenable algorithms that respect the non-commutative multiplication of Ore algebras has not been explored in a structured manner. By constructing skew resultant matrices with regular structure and by integrating modular and evaluation–interpolation steps, the present study lays the groundwork for parallel implementations of non-commutative elimination algorithms, even though full GPU realisations lie beyond the immediate scope of this thesis.

2.2 Cryptographic Constructions and Motivation

In this section, the foundational principles and ongoing development of both secret sharing and key exchange protocols are reviewed to establish the cryptographic context for this study. The discussion begins by examining the underlying structure of classical (t, n)-threshold schemes, which serve as the baseline for secure data distribution. Subsequently, the broad range of modern applications is highlighted, demonstrating the role of these primitives in today's systems.

However, significant attention is also given to the limitations and vulnerabilities inherent in traditional interpolation-based methods, particularly regarding adversary detection. This analysis leads to the motivation for a methodological shift towards a resultant-based algebraic framework, designed to provide inherent verification and enhanced security. Finally, existing Diffie–Hellman-like key-exchange protocols are reviewed to position the proposed skew-resultant framework within the post-quantum cryptographic domain.

2.2.1 Foundations of Secret Sharing

The rapid growth of emerging technologies such as artificial intelligence (AI), post-quantum computing, cloud computing, and secure multiparty computation has led to an unprecedented increase in the volume of digital data generated and processed globally. This expansion has been accompanied by a corresponding rise in security threats, ranging from large-scale data breaches to increasingly sophisticated cryptographic attacks. In this environment, strong data protection methods are essential to safeguard confidentiality, integrity, and availability of sensitive information. Cryptography therefore plays a central role and must continuously adapt and evolve to address these emerging challenges in a formal and mathematically provable manner.

Within this broad framework, *secret sharing* has become a fundamental cryptographic primitive for distributing sensitive information in a controlled way. Secret sharing was independently introduced by Shamir [120] and Blakley [8]. In Shamir's scheme, a secret is embedded as the constant term of a polynomial over a finite field, and shares are defined as evaluations of this polynomial at distinct points. Reconstruction requires a sufficient number of such evaluations, typically via polynomial interpolation. Blakley's approach, in contrast, employs a geometric framework in which the secret is represented as the intersection of hyperplanes in a multi-dimensional space, and shares correspond to points on these hyperplanes. Despite their structural differences, both constructions enforce that

only authorised subsets of participants can recover the secret, while unauthorised subsets gain no information.

These classical schemes provide a clean unconditional security reference point, but they were not designed with modern adversarial models (e.g., malicious insiders, adaptive corruption, or post-quantum attackers) in mind. They also embed no built-in verification for detecting misbehaving participants during reconstruction. Subsequent work on secret sharing has therefore aimed not only to generalise access structures but also to incorporate verification and robustness features. The present study contributes to this line of research by constructing resultant-based schemes that preserve unconditional security while adding inherent algebraic consistency checks and built-in adversary detection capabilities.

2.2.2 Classical (t, n)-Threshold Schemes

Secret sharing schemes in general, and (t, n)-threshold schemes in particular, have been extensively studied and surveyed in the literature, notably in [3, 23]. In a (t, n)-threshold scheme, a secret is distributed among n participants such that any subset of at least t participants can reconstruct the secret, whereas any subset of fewer than t participants learns nothing about it. This threshold framework is widely adopted because it simultaneously ensures data availability (even if some shares are lost or some participants are unavailable) and strict confidentiality that preventing unauthorised reconstruction. As discussed in [26], (t, n)-threshold schemes serve as building blocks in multiple higher-level cryptographic protocols.

Much of the classical work has focused on polynomial-based threshold schemes. The standard Shamir construction [120] uses univariate polynomials, whereas subsequent work has explored the use of bivariate polynomials to enable richer functionalities such as pairwise key agreement or more flexible access structures. In particular, Blundo et al. [9] used bivariate polynomials to derive pairwise keys between participants without requiring secure channels, while Harn et al. [72], Hsu et al. [76], and Liu et al. [96] employed asymmetric or symmetric

bivariate polynomials to secure communications among threshold-based groups. These constructions all rely on classical Shamir-style polynomial techniques as their underlying algebraic mechanism.

However, the heavy reliance on Shamir-style interpolation has several consequences. Firstly, most schemes retain a reconstruction process that assumes honest behaviour of all participants contributing shares; verification mechanisms, when present, are often add-on components rather than an integral part of the algebraic design. Secondly, these schemes typically reside entirely in the commutative polynomial setting and do not exploit richer algebraic structures such as skew polynomials or resultants to encode and check consistency. The resultant-based approach proposed in this thesis addresses these limitations by embedding the secret into the resultant of carefully constructed bivariate polynomials, so that algebraic properties of the resultant can be used both for secret recovery and for systematic verification.

2.2.3 Applications of Secret Sharing

The versatility and importance of secret sharing are evident from its broad range of cryptographic applications. Secret sharing schemes have been deployed in *secure data storage and management*, where sensitive data is split and distributed across multiple servers or devices to mitigate the risk of compromise of a single point of failure [133]. In *blockchain systems*, CHURP (CHUrn-Robust Proactive secret sharing) [98] uses bivariate polynomials for efficient and robust key management in highly dynamic environments, while Han et al. [69] combine blockchain with Shamir's scheme to design cloud storage architectures that offer both data confidentiality and verifiable data integrity.

Secret sharing also plays a central role in *secure multiparty computation* (MPC). Cramer et al. [41] provided efficient constructions for verifiable secret sharing and MPC, forming a foundation of many modern MPC protocols. More recently, Liu et al. [94] used secret sharing techniques to enable privacy-preserving

peer-to-peer energy trading, illustrating how secret sharing supports complex, real-world applications that involve sensitive commercial data. Beyond data storage and MPC, secret sharing is integral to *electronic voting systems*, where it helps protect the confidentiality and integrity of votes [7]. In *visual cryptography*, secret sharing techniques are adapted to image encryption, where images are split into meaningless shares that visually reveal the original only when correctly combined [125]. In the field of *digital signatures*, secret sharing can protect private signing keys by splitting them among multiple entities; for example, Ding et al. [45] proposed employing secret sharing to secure the SM2 private key for distributed signature generation.

While these applications demonstrate the breadth of secret sharing, they also highlight a typical pattern; most systems treat the underlying secret-sharing primitive as a black box with limited built-in support for adversary detection. Verifiable secret sharing and consistency checks are often handled at the protocol level rather than at the algebraic level of the scheme itself. This means that subtle algebraic vulnerabilities in the underlying sharing techniques propagate into higher-level applications. In contrast, the resultant-based schemes proposed in this thesis aim to integrate verification and adversary identification directly into the algebraic core, thus providing a more transparent foundation for applications such as storage and key management.

2.2.4 Vulnerabilities in Existing Schemes

Despite significant progress, several recent works have demonstrated that secret sharing schemes can still suffer from serious vulnerabilities. A recurring issue arises when the number of participants involved in reconstruction exceeds the minimum threshold t. In such settings, adversaries may compromise reconstruction groups and submit malicious shares, potentially recovering the secret or disrupting the protocol.

Harn et al. [70] proposed a scheme employing classical bivariate polynomials

within a threshold framework. Cheng et al. [30] subsequently showed that this approach is vulnerable; under certain conditions, an adversary can reconstruct the secret with fewer than the declared threshold number of shares. Similarly, Ahmadian and Jamshidpour [78] demonstrated that schemes by Harn and Hsu [71] can be compromised if an adversary participates in a reconstruction phase containing more than $t + 1$ participants. Ding et al. [46] showed that the scheme proposed in [105] is insecure because the secret can be reconstructed from specific combinations of publicly released shares.

More recently, Yang et al. [131] employed resultant techniques to analyse arithmetic-oriented (AO) cryptographic primitives, showing that algebraic tools such as resultants can, in some settings, be used to reduce the complexity of solving associated polynomial systems. Their results illustrate that algebraic elimination methods may become a powerful tool in the hands of an attacker if schemes are not carefully designed.

Taken together, these findings indicate that many existing constructions, particularly those based on classical bivariate polynomials and threshold mechanisms, may not adequately control the algebraic structure of their shares and reconstruction equations. They often lack robust methods for detecting inconsistent shares or identifying misbehaving participants.

These observations highlight the persistent need for schemes that not only distribute secrets securely but also incorporate *built-in verification* and *adversary detection* capabilities. The resultant-based schemes developed in this thesis address this need by enforcing that valid shares must arise from bivariate polynomials whose resultant satisfies carefully prescribed degree and constant-term conditions. Inconsistencies with these conditions can be detected through algebraic checks, and the structure of the resultant matrix can be exploited to pinpoint inconsistent shares, thus enabling adversary identification rather than mere detection of failure.

2.2.5 Beyond Classical Polynomial Methods

Most existing secret sharing schemes rely implicitly or explicitly on classical polynomial tools and associated algebraic properties for share generation, distribution, and reconstruction. In this context, polynomial *resultants* offer a natural but unexplored avenue for encoding and verifying relationships between polynomials. Classical resultants, as employed in elimination theory and computer algebra, have been studied extensively in commutative settings and, more recently, in skew (Ore) polynomial contexts [111, 112, 113, 114]. Yet the cryptographic literature has largely overlooked resultants as a design tool for secret sharing schemes.

This study explores the use of both classical [112] and skew [114] polynomial resultants as a foundation for constructing novel secret sharing schemes. To the best of current knowledge, this is the first work to employ resultants directly in the design of secret sharing schemes, thus opening a new line of research at the intersection of elimination theory and cryptography. The approach begins by identifying those classical resultant properties that are particularly suitable for cryptographic purposes such as degree control, constant-term encoding, and scaling behaviours, then uses them to construct resultant-based schemes with built-in verification checks. The work is subsequently extended to skew polynomials (a subclass of Ore polynomials [111]) in the non-commutative setting, where classical properties must be carefully generalised and adapted.

By leveraging bivariate (and, potentially, multivariate) polynomials and their resultants, the proposed constructions avoid the vulnerabilities identified in [70, 71, 105]. In particular, the schemes developed in this study are designed to be verifiable, and, when adversarial behaviour is detected during verification, the resultant-based structure provides mechanisms through which honest participants can identify inconsistent shares and thus isolate adversaries. This yields enhanced security and reliability compared with more traditional schemes that lack explicit adversary detection mechanisms and demonstrates how algebraic elimination can be used constructively rather than as a cryptanalytic threat.

2.2.6 From Interpolation-Based to Resultant-Based Secret Sharing

In the majority of secret sharing schemes, secret reconstruction is based on *Lagrange* or *Newton* interpolation [54, 119]. These polynomial interpolation techniques are algebraically natural and computationally efficient, but they tightly couple scheme correctness to the polynomial structure of the shares. As a consequence, security analyses often focus on the combinatorial aspects of threshold access structures, while the algebraic structure of the reconstruction function remains relatively static. To diversify this design space, alternative reconstruction methods have been explored, including those based on the *Chinese Remainder Theorem* (CRT) [81] and *Euler's Theorem* [25]. CRT-based schemes exploit residue-class representations and can offer different performance and robustness trade-offs, but they introduce additional arithmetical structure (e.g., moduli selection and co-primality conditions) that must be carefully managed to avoid information leakage. Schemes based on Euler's theorem share similar sensitivities and, in practice, may be less flexible for dynamic threshold adjustment or for integration with non-commutative structures. The present study contributes to this line of work by employing resultants of bivariate polynomials (and their skew analogues) as the primary algebraic device for both secret encoding and verification. In contrast to classical interpolation-based schemes, where reconstruction is a matter of polynomial interpolation at known points, the resultant-based approach encodes the secret into the determinant-like object associated with a Sylvester matrix. This enables verification conditions based on degree bounds, constant-term behaviour, and evaluation properties that are not readily expressed in interpolation-only frameworks.

2.2.7 Post-Quantum Directions

In the context of post-quantum cryptography, *lattice-based threshold secret sharing* schemes [26] have emerged as a leading candidate class, deriving security

from numerical lattice problems (such as Learning With Errors or Shortest Vector Problem) that are believed to be hard for both classical and quantum adversaries [107]. However, these approaches often incur substantial computational and storage overheads, require delicate parameter selection, and typically rely on numerical shares (e.g. vectors of integers). Furthermore, achieving verifiability in lattice settings often necessitates complex external mechanisms, such as zero-knowledge proofs, which can complicate the protocol structure. The present study investigates an alternative framework, based on polynomial resultants, which diverges significantly from conventional lattice-based PQC.

Fundamentally, this approach employs *Masked Symbolic Shares of Bivariate Polynomials* as the primary cryptographic primitive. Consequently, security is grounded not in lattice assumptions, but in the complexity of problems in polynomial algebra over a chosen field, thus avoiding the specific numerical structures (such as Factoring and DLP) vulnerable to Shor's algorithm. A distinct advantage of this algebraic formulation is its capacity for *built-in, dealer-facilitated* verification. Unlike schemes that require add-on verification protocols, the proposed method leverages inherent resultant properties (specifically scaling and evaluation) to implement robust consistency checks. This allows participants to detect inconsistencies and identify adversaries collaboratively, integrating verification directly into the algebraic core of the reconstruction process without requiring further involvement from the dealer.

2.2.8 Key-Exchange via Skew Resultant

In this section, the investigation extends to the domain of key-exchange protocols constructed over non-commutative algebraic structures. The discussion begins by reviewing early attempts to utilise univariate skew polynomial rings for Diffie–Hellman-like primitives and the subsequent cryptanalytic that exploited their Euclidean properties. Following an analysis of mitigation strategies based on commuting subsets in multivariate settings, the section leads to the proposal

of a novel protocol. This new approach leverages the inherent commutativity of skew resultants to establish a shared secret, thereby addressing vulnerabilities while avoiding the complexity of artificial commuting substructures.

2.2.8.1 Non-Commutative Structures and Early Protocols

The search for post-quantum cryptographic primitives has also motivated research into key-exchange protocols based on non-commutative algebraic structures. The intuition is that non-commutativity introduces algebraic complexity that offer resistance to both classical and quantum attacks. Boucher et al. [11] proposed a Diffie–Hellman-like protocol constructed over a univariate skew (Ore) polynomial ring. In their design, public keys are obtained by evaluating polynomials under a non-commutative composition, and security is intended to derive from the difficulty of reversing these compositions.

However, Dubois and Kammerer [48] demonstrated that the specific ring used in [11] is a Euclidean domain. This structural property enabled them to perform greatest common divisor (GCD) computations that effectively recovered the private keys, thus rendering the scheme insecure. This attack illustrates that non-commutativity alone is insufficient; one must carefully consider additional ring-theoretic properties such as Euclidean property and the existence of efficient GCD algorithms. Thus, the choice of the underlying Ore domain in [11] proved insufficient to prevent commutative-like attacks, revealing that non-commutative constructions can degenerate into more tractable structures if their algebraic parameters are not chosen carefully.

2.2.8.2 The Commuting Subset Approach

In response to the weaknesses identified in the previous subsection, Burger and Heinle [18] suggested moving to *multivariate* Ore polynomial rings, which are generally not Euclidean domains and hence are not vulnerable to the GCD-based attack of [48]. Their protocol, however, requires participants to select private keys

from specially constructed subsets of commuting polynomials embedded within the larger non-commutative ring. While this design avoids some of the structural weaknesses of earlier schemes, it introduces new practical and theoretical challenges. The construction of these commuting subsets can be technically involved and computationally expensive, and the very structure of these subsets may become an additional attack surface if an adversary can exploit algebraic relations among commuting elements. Moreover, the security of the protocol becomes dependent on the hardness of problems restricted to these commuting subfamilies, rather than on the full non-commutative structure of the underlying Ore algebra.

These issues highlight a broader gap; how to design key-exchange or related cryptographic protocols in non-commutative settings without relying on artificial or vulnerable commuting substructures, and without reintroducing vulnerabilities through hidden Euclidean properties or linearisation techniques. There is a need for constructions that genuinely use non-commutative structure in a more inherent way and for hardness assumptions that are less sensitive to unintended algebraic simplifications.

2.2.8.3 Skew Resultants as an Inherent Source of Commutativity

A central gap identified in the preceding discussion is the lack of an inherent, algebraically structured source of commutativity in non-commutative (Ore) settings for cryptographic use without resorting to vulnerable Euclidean domains or artificial constructions.

The present study addresses this gap by investigating *skew resultants* as an inherent, yet largely unexplored, source of structured commutativity within Ore algebras. Skew resultants, as developed in recent work [113, 114], extend classical resultant theory to skew polynomial rings, providing algebraic objects that encode common solutions of operator polynomials while maintaining compatibility with non-commutative multiplication. Rather than imposing commuting subsets manually via explicit construction, the proposed approach exploits the natural

structure afforded by skew resultants to derive shared cryptographic material (such as keys or verification values) from appropriately constructed polynomial systems.

Within this framework, the resultant-based intuition that underpins the secret sharing schemes presented herein can be adapted to key-exchange settings. Although the detailed key-exchange protocol is discussed in later chapters (Chapter 8) and lies beyond the scope of this subsection, the central motivation is clear, namely by working directly with skew resultants in non-commutative rings; one may obtain protocols that avoid Euclidean-domain vulnerabilities, reduce reliance on artificial commuting subsets, and offer alternative, algebraically rich hardness assumptions potentially distinct from those underpinning classical or lattice-based schemes. In doing so, this work positions skew resultants not only as an effective algebraic tool for elimination but also as a promising building block for future non-commutative and potentially post-quantum cryptographic protocols that integrate key exchange, secret sharing, and verification within a unified algebraic framework.

2.2.8.4 Implications of Algebraic Attacks and the Role of Non-Commutativity

From a cryptanalytic perspective, the primary threat to the proposed protocol lies in algebraic attacks where an adversary attempts to reconstruct private factors from the public values. Unlike univariate schemes susceptible to GCD-based attacks [48], the multivariate Ore setting employed here lacks a Euclidean structure, thereby preventing standard greatest common divisor exploits. Furthermore, generic linearisation or factorisation attacks face significant computational barriers; while commutative systems are often amenable to Gröbner basis techniques, the non-commutative nature of Ore multiplication introduces significant algorithmic complexity, severely limiting the efficacy of existing solvers [42] Consequently, the protocol relies on the hardness of the Skew Resultant Conjugacy

Problem (SRCP). By leveraging the *inherent* commutativity of skew resultants derived from the Dieudonné determinant, the design avoids the structural weaknesses and computational bottlenecks associated with the artificially constructed commuting subsets used in earlier proposals [18]. However, non-commutativity alone is not a sufficient defence; robust parameter selection remains essential to ensure the algebraic structure does not degenerate into a form susceptible to low-dimensional representation attacks.

Chapter 3

Algebraic Preliminaries for Ore Polynomials and Resultants

The subsequent developments in this study rely on a unified algebraic framework for working with non-commutative polynomial structures. In particular, the resultant constructions and their cryptographic applications are formulated in the setting of Ore (or skew) polynomial rings, where coefficients lie in a (skew) field and multiplication is twisted by a ring endomorphism and a σ-derivation. This chapter assembles the necessary background on rings, modules, ideals, Ore extensions, and determinant theory in the non-commutative context, and lays the groundwork for the definition and analysis of skew resultants in later chapters.

Unless otherwise stated, throughout this study $\mathcal{F}$ denotes a (skew) field (or division ring): a unital associative (not necessarily commutative) ring in which every non-zero element admits a multiplicative inverse, so that $\mathcal{F}$ is a domain. The symbol $\mathcal{R}$ denotes a (not necessarily commutative) ring, often assumed to be a domain when divisibility or factorisation are considered. After recalling basic notations for left, right, and two-sided ideals and their finitely generated and principal variants, the notion of left and right $\mathcal{R}$-modules is reviewed, as these provide the natural environment for interpreting Ore polynomials as operators and for formulating the concepts of similarity and factorisation in non-commutative

settings.

A central theme of this chapter is the construction and use of Ore polynomial rings $\mathcal{R}[\theta; \sigma, \delta]$, where σ is an endomorphism of $\mathcal{R}$ and δ is a σ-derivation satisfying

$$\delta(ab) \;=\; \sigma(a)\,\delta(b) + \delta(a)\,b, \qquad a, b \in \mathcal{R}.$$

The resulting skew-polynomial structure is characterised by the commutation rule

$$\theta a \;=\; \sigma(a)\,\theta + \delta(a), \qquad a \in \mathcal{R},$$

which induces non-trivial interactions between the indeterminate and the coefficients. Special attention is paid to the case where $\mathcal{R} = \mathcal{F}$ is a skew field and σ is an automorphism, as $\mathcal{F}[\theta; \sigma, \delta]$ is then both a principal left ideal domain (PLID) and a principal right ideal domain (PRID). This property ensures the existence of *Euclidean algorithms*, greatest common right (and left) divisors, and least common left (and right) multiples, all of which are used later in the analysis of resultants.

To support the non-commutative determinant constructions needed for skew resultants, the chapter recalls the Dieudonné determinant for matrices over a skew field. Unlike the classical determinant, the Dieudonné determinant takes values in an abelian quotient of the multiplicative group (specifically, the group modulo its commutator subgroup). Nonetheless, it satisfies analogues of the key multiplicative properties and row/column operations familiar from the commutative case. Particular emphasis is placed on situations where the non-commutative coefficient ring admits an embedding into a skew field of fractions (via the Ore condition), and on almost-commutative settings where filtered rings and associated graded rings allow the principal symbol of the Dieudonné determinant to be interpreted more canonically.

Another important ingredient is the notion of similarity in non-commutative domains and its connection with unique factorisation. Two elements $f, g \in \mathcal{R}$ are

termed similar if the corresponding quotient left $\mathcal{R}$-modules $\mathcal{R}/_{\mathcal{R}}\langle f\rangle$ and $\mathcal{R}/_{\mathcal{R}}\langle g\rangle$ are isomorphic. In PIDs, including $\mathcal{F}[\theta;\sigma,\delta]$, similarity provides the natural non-commutative analogue of *association* in commutative unique factorisation domains. This leads to the concept of a non-commutative unique factorisation domain ($UFD_{\approx}$), where factorisations into irreducibles are unique up to order and similarity of the factors. The relevance of this framework is that it allows the factorisation properties of Ore polynomials to be controlled in a manner compatible with the resultant constructions.

The chapter also formalises the passage from univariate to multivariate settings via *iterated Ore extensions* (often termed multivariate Ore algebras), of the form

$$\mathcal{S} = \mathcal{F}[\theta_1;\sigma_1,\delta_1]\,[\theta_2;\sigma_2,\delta_2]\cdots[\theta_n;\sigma_n,\delta_n],$$

with particular attention to the case where all $\delta_i = 0$, each σ_i is an automorphism, and the indeterminates θ_i commute. Under suitable compatibility conditions on the σ_i, the resulting ring admits a well-behaved notion of degree and supports generalised Sylvester-type constructions.

The bivariate skew polynomial ring $\mathcal{F}[\theta_1;\sigma_1][\theta_2;\sigma_2]$ (assuming $\delta_1 = \delta_2 = 0$) emerges as the main algebraic setting for the skew resultants studied in subsequent chapters; many of the constructions and proofs are carried out in this iterated bivariate Ore algebra and rely on structural properties of non-commutative rings, including the theory of skew (Ore) polynomials, their divisibility and ideal structure, and the way in which resultants interact with these features.

In summary, this chapter assembles the algebraic tools required for the non-commutative resultant theory developed later. It introduces the language of ideals and modules, defines Ore polynomial rings and their operator actions, describes Euclidean and principal ideal properties, recalls the Dieudonné determinant, and formalises similarity and non-commutative unique factorisation. These ingredients form the algebraic backbone for the Sylvester-type matrices, skew resultants, and elimination techniques that will be constructed and analysed in the subse-

quent chapters.

3.1 Basic Notations for Ideals and Modules

This section briefly recalls some relevant notations and concepts of ideals and modules within the non-commutative context, which is central to this study.

An additive subgroup I of a ring $\mathcal{R}$ is termed a *left* (*resp. right*) *ideal* if $ra \in I$ (resp. $ar \in I$) for all $a \in I$ and $r \in \mathcal{R}$. If I is both a left and a right ideal, it is called a *two-sided ideal*. If a left (resp. right) ideal I is generated by a finite set of elements $a_1, \ldots, a_n \in \mathcal{R}$, then I is termed *finitely generated* and denoted by ${}_{\mathcal{R}}\langle a_1, \ldots, a_n\rangle$ (resp. $\langle a_1, \ldots, a_n\rangle_{\mathcal{R}}$). Similarly, a two-sided ideal generated by finite elements $a_1, \ldots, a_n \in \mathcal{R}$ is denoted by ${}_{\mathcal{R}}\langle a_1, \ldots, a_n\rangle_{\mathcal{R}}$, which in this case can also be written as $(a_1, \ldots, a_n)$. A special case arises when a left (resp. right) ideal I can be generated by a single element, say $a \in I$; such an ideal is termed *left (resp. right) principal*. Furthermore, a two-sided ideal I generated by a single element $a \in I$ such that $\mathcal{R}a = a\mathcal{R} = I$ is called a *principal ideal*. If every left (resp. right) ideal of a ring $\mathcal{R}$ is principal, then $\mathcal{R}$ is called a *left* (*resp. right*) *principal ideal ring*. If $\mathcal{R}$ is also a domain, it is called a *left* (*resp. right*) *principal ideal domain* (abbreviated as left (resp. right) PID). A left (resp. right) PID is also *left (resp. right) Noetherian*, that is, every its left (resp. right) ideal is finitely generated.

A *left $\mathcal{R}$-module* M is an abelian group $(M, +)$ equipped with an operation $\cdot : \mathcal{R} \times M \to M$, written as $r \cdot m$ or simply rm for $r \in \mathcal{R}$ and $m \in M$, satisfying the following for all $r, s \in \mathcal{R}$ and $m, n \in M$:

1. $r(m + n) = rm + rn$;
2. $(r + s)m = rm + sm$;
3. $(rs)m = r(sm)$;
4. $1_{\mathcal{R}}m = m$, where $1_{\mathcal{R}}$ is the multiplicative identity in $\mathcal{R}$.

A *right $\mathcal{R}$-module M* is defined analogously, with the ring $\mathcal{R}$ acting on the right; for example, property 3 becomes $(ms)r = m(sr)$. A left $\mathcal{R}$-module and a right $\mathcal{R}$-module is referred to simply as an *$\mathcal{R}$-module* (or sometimes just a *bimodule over $\mathcal{R}$*)

Key Non-Commutative Differences: Left vs. Right Ideals

In a non-commutative ring, the *side* on which multiplication acts matters. A subset may be closed under multiplication from the left (*left ideal*) without being closed under multiplication from the right (*right ideal*). The distinction is illustrated below for an Ore algebra.

Example 3.1.1. *Let $\mathcal{S} = \mathbb{Q}[x][\partial; \sigma, \delta]$ denote the Ore algebra of differential operators (first Weyl algebra), formulated as a skew polynomial ring over the base ring $\mathcal{F} = \mathbb{Q}[x]$. Here, $\sigma = \mathrm{id}$ is the identity automorphism and $\delta = \frac{d}{dx}$ is the standard derivation. The non-commutative multiplication rule is given by the commutation relation:*

$$\partial x = \sigma(x)\partial + \delta(x) = x\partial + 1.$$

Consider the principal left ideal generated by ∂, denoted as $L = \mathcal{S}\,\partial$. By construction, for any skew polynomial $P \in \mathcal{S}$, the product $P \cdot \partial$ remains in L. Thus, L is a left ideal.

Now consider $\partial \in L$ and $x \in \mathcal{S}$. Applying the commutation rule yields:

$$\partial \cdot x = x\partial + 1.$$

The resulting element $x\partial + 1$ contains a term of degree 0 in ∂ (the constant 1). However, every element in L is of the form $P\partial$ and thus must have a degree in ∂ of at least 1 (assuming $P \neq 0$). Consequently, $x\partial + 1 \notin L$, demonstrating that L is not a right ideal.

3.2 Pseudo-Linear Maps

Before introducing Ore polynomial rings formally, the concept of *pseudo-linear maps* is reviewed. These maps describe the action of the indeterminates in Ore constructions and are fundamental to the study of linear differential and difference operators. These definitions found in works such as [15, 89].

Definition 3.2.1 (σ-Derivation)**.** *Let $\mathcal{R}$ be a ring and σ be an endomorphism of $\mathcal{R}$. An additive map $\delta : \mathcal{R} \to \mathcal{R}$ that satisfies*

$$\delta(ab) = \sigma(a)\delta(b) + \delta(a)b, \quad \forall a, b \in \mathcal{R} \tag{3.1}$$

is called a σ-derivation (or pseudo-derivation) of $\mathcal{R}$.

A more general case is the pseudo-linear map of modules:

Definition 3.2.2 ($\mathcal{F}$-Pseudo-Linear Map)**.** *Consider a left $\mathcal{F}$-module $\mathcal{V}$ (where $\mathcal{F}$ is a skew field), an endomorphism σ of $\mathcal{F}$, and a σ-derivation δ of $\mathcal{F}$. An additive map $\varphi : \mathcal{V} \to \mathcal{V}$ is called $\mathcal{F}$-pseudo-linear if*

$$\varphi(a\mathbf{u}) = \sigma(a)\varphi(\mathbf{u}) + \delta(a)\mathbf{u}, \quad \forall a \in \mathcal{F} \text{ and } \forall \mathbf{u} \in \mathcal{V}. \tag{3.2}$$

Definition 3.2.3 (Constant Elements)**.** *The set of all constant elements of $\mathcal{F}$ with respect to σ and δ is*

$$\mathrm{Const}(\mathcal{F}) = \{a \in \mathcal{F} \mid \sigma(a) = a,\ \delta(a) = 0\},$$

which forms a division subring (or skew subfield) of $\mathcal{F}$. If $\mathcal{F}$ is commutative, $\mathrm{Const}(\mathcal{F})$ *is a subfield.*

From this definition, the following property, noted by [15], can be concluded:

Lemma 3.2.4. *Any $\mathcal{F}$-pseudo-linear map is* $\mathrm{Const}(\mathcal{F})$*-linear.*

Proof. Let $\varphi : \mathcal{V} \to \mathcal{V}$ be an $\mathcal{F}$-pseudo-linear map (with respect to fixed structure maps $\sigma : \mathcal{F} \to \mathcal{F}$ and $\delta : \mathcal{F} \to \mathcal{F}$), meaning that φ is additive and satisfies

$$\varphi(av) = \sigma(a)\,\varphi(v) + \delta(a)\,v \qquad \text{for all } a \in \mathcal{F},\ v \in V.$$

From Definition 3.2.3:

$$\mathrm{Const}(\mathcal{F}) \ = \ \{\, a \in \mathcal{F} \mid \sigma(a) = a \text{ and } \delta(a) = 0 \,\}.$$

Fix $a \in \mathrm{Const}(\mathcal{F})$ and $v \in V$. Then, by pseudo-linearity;

$$\varphi(av) = \sigma(a)\,\varphi(v) + \delta(a)\,v = a\,\varphi(v) + 0 = a\,\varphi(v).$$

□

Together with additivity of φ, this shows that φ respects scalar multiplication by elements of $\mathrm{Const}(\mathcal{F})$, hence φ is $\mathrm{Const}(\mathcal{F})$-linear.

These relationships (involving σ and δ) highlight the close connection between pseudo-linear maps, derivations, and automorphisms which support the structure of Ore polynomial rings described below.

3.3 Ore Polynomial Rings

Ore polynomial rings offer a powerful framework for modelling a wide range of linear functional systems, including those involving linear differential and difference operators. A key advantage of this framework is that it allows for the generic study and implementation of algebraic operations that can then be applied to each specific context. The formal definition is as follows [111].

Definition 3.3.1 (Ore Polynomial Ring)**.** *Let $\mathcal{R}$ be a ring, σ an endomorphism of $\mathcal{R}$, and δ a σ-derivation on $\mathcal{R}$. The* Ore polynomial ring *(or* skew polynomial ring*), denoted $\mathcal{R}[\theta; \sigma, \delta]$, is the set of polynomials in an indeterminate θ with*

coefficients written on the left:

$$\mathcal{R}[\theta;\sigma,\delta] = \left\{ \sum_{i=0}^{n} c_i \theta^i \mid c_i \in \mathcal{R}, n \in \mathbb{N}_0 \right\}.$$

Addition in this set is defined component-wise. Multiplication is uniquely determined by extending the multiplication in $\mathcal{R}$ via the distributive law, subject to the commutation rule:

$$\theta a = \sigma(a)\theta + \delta(a), \quad \forall a \in \mathcal{R}. \tag{3.3}$$

The resulting algebraic structure is an associative ring containing $\mathcal{R}$ as a subring.

If $\mathcal{R}$ is a domain and σ is injective, then $\mathcal{R}[\theta;\sigma,\delta]$ is a domain [65, p. 37]. If $\mathcal{R} = \mathcal{F}$ is a (skew) field and σ is an automorphism, $\mathcal{F}[\theta;\sigma,\delta]$ is a principal left ideal domain (PLID) and a principal right ideal domain (PRID), hence it is Noetherian. Elements of $\mathcal{F}[\theta;\sigma,\delta]$ are termed (univariate) Ore polynomials or (univariate) Ore operators. This study primarily considers left Ore polynomials over a (skew) field $\mathcal{F}$, where coefficients are written to the left of the powers of θ. If $\mathcal{F}$ is a $\mathcal{K}$-algebra for some field $\mathcal{K}$ (often $\mathcal{K} \subseteq \mathrm{Const}(\mathcal{F})$), and σ, δ are $\mathcal{K}$-linear, then θ commutes with elements of $\mathcal{K}$.

A non-zero Ore polynomial $f \in \mathcal{R}[\theta;\sigma,\delta]$ has degree n (denoted $\deg(f) = n$) if $n \geq 0$ is the largest integer such that the coefficient of θ^n is non-zero. This coefficient is the leading coefficient, $\mathrm{lc}(f)$. The degree of the zero polynomial is $-\infty$.

Remark 3.3.2 (Degree Properties)**.** *For $f, g \in \mathcal{R}[\theta;\sigma,\delta]$:*

(i) $\deg(f+g) \leq \max(\deg(f), \deg(g))$.

(ii) If $\mathcal{R}$ is a domain and σ is injective, $\deg(fg) = \deg(f) + \deg(g)$.

Since this study usually assumes $\mathcal{F}$ is a (skew) field and σ is an automorphism (hence injective), property (ii) typically hold for $\mathcal{F}[\theta;\sigma,\delta]$.

Due to their non-commutative structure, Ore polynomials are usually studied

with a convention of either consistently placing coefficients to the left of the indeterminate (a *left Ore polynomial*) or to the right (a *right Ore polynomial*).

It is generally possible to develop the theory of Ore polynomials starting from either convention and obtain equivalent algebraic properties, especially if the defining endomorphism σ is an automorphism. This allows for conversion between left and right polynomial forms. For instance, if σ is an automorphism, a left Ore polynomial $a_1\theta + a_0$ (where coefficients $a_1, a_0 \in \mathcal{R}$) can be rewritten as an equivalent right Ore polynomial $\theta\sigma^{-1}(a_1) + (a_0 - \delta(\sigma^{-1}(a_1)))$. This transformation relies on the commutation rule (Equation 3.3 in the context of left polynomials, $\theta r = \sigma(r)\theta + \delta(r)$, implies $r\theta = \theta\sigma^{-1}(r) - \delta(\sigma^{-1}(r))$ if σ is an automorphism). For a detailed treatment of such conversions (see, for example, [17, Proposition 3.9]).

Once a convention is chosen, say left Ore polynomials, any such polynomial f in $\mathcal{R}[\theta; \sigma, \delta]$ can be uniquely written in the canonical form $f = \sum_{i=0}^{n} c_i\theta^i$, where $c_i \in \mathcal{R}$. In this study, the focus is primarily on left Ore polynomials over a (skew) field $\mathcal{F}$.

If $\mathcal{F}$ is a $\mathcal{K}$-algebra, where $\mathcal{K}$ is a field (usually central or consisting of constants with respect to σ and δ), then the $\mathcal{K}$-linearity of σ and δ ensures that the indeterminate θ commutes with each element in $\mathcal{K}$.

As mentioned earlier, this model benefits significantly from the ability to define and implement operations in a generic and uniform manner, and then suitably apply them to each specific instance.

The following are two typical examples illustrating Ore polynomial rings constructed with differential and shift operators.

Example 3.3.3. If D is a derivation operator on a ring $\mathcal{R}$ (meaning $D(rs) = rD(s) + D(r)s$ for all $r, s \in \mathcal{R}$), then the Ore polynomial ring $\mathcal{R}[\theta; \mathrm{id}, D]$ is the ring with multiplication defined by the rule:

$$\theta r = \mathrm{id}(r)\theta + D(r) = r\theta + D(r), \quad \text{for all } r \in \mathcal{R}.$$

This ring is isomorphic to the ring of linear homogeneous differential operators in one differential indeterminate θ over $\mathcal{R}$ and is an instance of an Ore polynomial ring of derivation type.

Example 3.3.4. Let E be an injective endomorphism of a domain $\mathcal{R}$, and let the associated derivation δ be the null map $\mathbf{0}$. Then the Ore polynomial ring $\mathcal{R}[\theta; E, \mathbf{0}]$ is defined by the multiplication rule:

$$\theta r = E(r)\theta + \mathbf{0}(r) = E(r)\theta, \quad \text{for all } r \in \mathcal{R}.$$

This ring is isomorphic to the ring of linear homogeneous shift operators (when E is a shift operator, e.g., $E(r(t)) = r(t+1)$) in one indeterminate θ (with respect to E) over $\mathcal{R}$. It is an Ore polynomial ring of automorphism type.

The following table (Table 3.1) provides some further examples of Ore operators, detailing the specific choices for σ, δ, and the resulting commutation rule for the indeterminate θ with elements $a(t)$ from the coefficient ring. Such a coefficient ring is often of the form $\mathcal{R} = \mathcal{K}(t)$, for some field $\mathcal{K}$ and a parameter t. For more examples, refer to [31].

Table 3.1: Common examples of Ore operator structures in the ring $\mathcal{K}(t)[\theta; \sigma, \delta]$.

Coefficient Ring	**Operator Type**	$\boldsymbol{\sigma(a(t))}$	$\boldsymbol{\delta(a(t))}$	**Rule for** $\boldsymbol{\theta t}$
$\mathcal{K}(t)$	Differential	$a(t)$	$a'(t)$	$\theta t = t\theta + 1$
	Difference	$a(t+1)$	$a(t+1) - a(t)$	$\theta t = (t+1)\theta + 1$
	Shift	$a(t+1)$	0	$\theta t = (t+1)\theta$
	Eulerian	$a(t)$	$ta'(t)$	$\theta t = t\theta + t$
$\mathcal{K}(q,t)$ $(q \in \mathbb{Q} \setminus \{0, 1, -1\})$	q-differential	$a(qt)$	$\frac{a(qt)-a(t)}{(q-1)t}$	$\theta t = qt\theta + 1$
	q-difference	$a(qt)$	$a(qt) - a(t)$	$\theta t = qt\theta + (q-1)t$
	q-shift	$a(qt)$	0	$\theta t = qt\theta$

Remark 3.3.5 (Skew Polynomial and Commutative Rings)**.** *The case when $\delta = 0$ in Definition 3.3.1 yields what is often termed a* skew polynomial ring of automorphism type*, denoted by $\mathcal{R}[\theta; \sigma]$. (It should be noted that the term* skew polynomial ring *may refer to different algebraic structures in some literature.) The* ordinary commutative polynomial ring *$\mathcal{F}[\theta]$ is a special instance of an Ore polynomial ring where $\mathcal{F}$ is a commutative field, σ is the identity map on $\mathcal{F}$, and $\delta = 0$. Consequently, results developed for Ore or skew polynomials presented in this work can be specialised to their commutative counterparts.*

Ore polynomial rings can be constructed iteratively to define rings with multiple indeterminates.

Definition 3.3.6 (Multivariate Ore Algebra (Iterated Ore Extension))**.** *Let $\mathcal{R}_0 = \mathcal{F}$ be a (skew) field. An iterated Ore extension is defined recursively: $\mathcal{R}_k = \mathcal{R}_{k-1}[\theta_k; \sigma_k, \delta_k]$ for $k = 1, \ldots, n$, where σ_k is an automorphism of $\mathcal{R}_{k-1}$ and δ_k is a σ_k-derivation on $\mathcal{R}_{k-1}$. A common specific type is the* Ore algebra *where $\mathcal{F}$ is a commutative field $\mathbb{Q}(t_1, \ldots, t_m)$, the θ_i commute with each other ($\theta_i\theta_j = \theta_j\theta_i$), σ_i and δ_j commute for $i \neq j$, and satisfy $\sigma_i(\theta_j) = \theta_j, \delta_i(\theta_j) = 0$ for $i > j$. This study focuses on a specific type of multivariate skew polynomial ring $\mathcal{S} = \mathcal{F}[\theta_1; \sigma_1][\theta_2; \sigma_2] \cdots [\theta_n; \sigma_n]$ where $\mathcal{F}$ is a (skew) field, all $\delta_i = 0$, the indeterminates θ_i commute with each other, and the automorphisms σ_i act on $\mathcal{F}$ and satisfy $\sigma_j(\theta_i) = \theta_i$ for $i \neq j$, and $\sigma_j\sigma_i = \sigma_i\sigma_j$. For such a ring, $\theta_i a = \sigma_i(a)\theta_i$ for $a \in \mathcal{F}$.*

The conditions specified in Definition 3.3.6 concerning the commuting nature of indeterminates θ_k, θ_l (for $k \neq l$) and the properties of the automorphisms σ_k, σ_l are interconnected and fundamental to the structure of these iterated skew polynomial rings. Specifically, the assumption that $\theta_k\theta_l = \theta_l\theta_k$ implies the condition $\sigma_k(\theta_l) = \theta_l$ (for $l \neq k$) when σ_k is extended to act on $\mathcal{R}_{l-1}[\theta_l; \sigma_l]$ (if $k > l$). For instance, considering the construction $\mathcal{R}_k = (\mathcal{R}_l)[\theta_k; \sigma_k]$ where θ_l is in $\mathcal{R}_l$, if θ_k and θ_l commute, then $\theta_l\theta_k = \theta_k\theta_l = \sigma_k(\theta_l)\theta_k$. A comparison of coefficients then indicates that $\sigma_k(\theta_l)$ must be equal to θ_l.

Furthermore, the property that the automorphisms σ_k and σ_l (for $k \neq l$) commute in their action on the base (skew) field $\mathcal{F}$ (i.e. $\sigma_k\sigma_l = \sigma_l\sigma_k$ when applied to any $a \in \mathcal{F}$) is also inherently linked to the commutativity of the indeterminates θ_k and θ_l. This can be observed by considering their successive actions on an arbitrary element $a \in \mathcal{F}$:

$$\theta_k\theta_l a = \theta_l\theta_k a. \tag{3.4}$$

The left-hand side of Equation (3.4) expands as: $\theta_k\theta_l a = \theta_k(\sigma_l(a)\theta_l) = \sigma_k(\sigma_l(a))\theta_k\theta_l$. The right-hand side expands as: $\theta_l\theta_k a = \theta_l(\sigma_k(a)\theta_k) = \sigma_l(\sigma_k(a))\theta_l\theta_k$. Given that the indeterminates θ_k and θ_l commute (i.e. $\theta_k\theta_l = \theta_l\theta_k$), equating the coefficients of a in these expanded forms necessitates that $\sigma_k(\sigma_l(a)) = \sigma_l(\sigma_k(a))$ for all $a \in \mathcal{F}$. Thus, the automorphisms σ_k and σ_l must commute in their action on $\mathcal{F}$. Conversely, if the automorphisms commute on $\mathcal{F}$ and satisfy the condition $\sigma_k(\theta_l) = \theta_l$ for $k \neq l$, the indeterminates θ_k and θ_l will indeed commute as defined. These conditions ensure a consistent and well-defined algebraic structure for the multivariate skew polynomial ring.

For instance, $(\mathcal{F}[\theta_1;\sigma_1])[\theta_2;\sigma_2]$ (with $\delta_1 = \delta_2 = 0$ and θ_1, θ_2 commuting as per the above conditions) is the bivariate skew polynomial ring, which serves as a primary initial focus in this study. Elements f of such a ring $\mathcal{F}[\theta_1;\sigma_1][\theta_2;\sigma_2]$ are typically written as polynomials in one indeterminate, say θ_2, whose coefficients are polynomials in the other indeterminate; for example, $f = \sum_{k=0}^{n} a_k(\theta_1)\theta_2^k$, where $a_k(\theta_1) \in \mathcal{F}[\theta_1;\sigma_1]$. The degree of a non-zero polynomial f with respect to θ_2, denoted by $\deg_{\theta_2}(f)$, is the highest power $n \geq 0$ of θ_2 with a non-zero coefficient $a_n(\theta_1)$, or $-\infty$ for the zero polynomial. The degree properties (as outlined in Remark 3.3.2 for the univariate case) extend naturally; for example, for $f, g \in \mathcal{F}[\theta_1;\sigma_1][\theta_2;\sigma_2]$, $\deg_{\theta_2}(fg) = \deg_{\theta_2}(f) + \deg_{\theta_2}(g)$, provided the leading coefficients with respect to θ_2 are not zero divisors (which is guaranteed if working over a skew field $\mathcal{F}$ and $\mathcal{F}[\theta_1;\sigma_1]$ is a domain).

Action of Ore Polynomials as Operators

Elements of an Ore polynomial ring can act as operators on suitable modules. This section outlines this action, primarily considering Ore polynomials over a (skew) field $\mathcal{F}$, consistent with the main focus of this study.

The set of additive endomorphisms of $\mathcal{F}$, denoted $\mathrm{End}_{\mathbb{Z}}(\mathcal{F})$, forms a ring where addition is pointwise and multiplication is composition: $(f+g)(a) = f(a)+g(a)$ and $(fg)(a) = f(g(a))$ for all $a \in \mathcal{F}$ and $f, g \in \mathrm{End}_{\mathbb{Z}}(\mathcal{F})$. The field $\mathcal{F}$ itself can be embedded into this endomorphism ring by identifying each element $c \in \mathcal{F}$ with the operator $c\,\theta^0$ (which as a polynomial is simply c), whose action is defined as left multiplication: $(c\,\theta^0)(b) = cb$ for all $b \in \mathcal{F}$.

Let $\varphi : \mathcal{F} \to \mathcal{F}$ be an $\mathcal{F}$-pseudo-linear map (as per Definition 3.2.2). An Ore polynomial $f(\theta) = \sum_{j=0}^{m} c_j \theta^j \in \mathcal{F}[\theta; \sigma, \delta]$ can be made to act on an element $\mathbf{u} \in \mathcal{F}$ by interpreting θ as the map φ. This action is defined as:

$$f(\varphi)(\mathbf{u}) = \sum_{j=0}^{m} c_j \varphi^j(\mathbf{u}), \quad \text{where } c_j \in \mathcal{F}, \mathbf{u} \in \mathcal{F}, \text{ and } \varphi^0(\mathbf{u}) = \mathbf{u}. \tag{3.5}$$

In this context, $f(\varphi)$ denotes the operator obtained by substituting the indeterminate θ in $f(\theta)$ with the map φ.

More generally, let $\mathcal{S} = \mathcal{F}[\theta; \sigma, \delta]$ and let $\mathcal{V}$ be a left $\mathcal{S}$-module. An $\mathcal{F}$-pseudo-linear map $\varphi\colon \mathcal{V} \to \mathcal{V}$ (compatible with σ and δ from $\mathcal{S}$) induces an action, denoted by $\bullet$, of $\mathcal{S}$ on $\mathcal{V}$:

$$\begin{aligned} &\mathcal{S} \times \mathcal{V} \to \mathcal{V} \\ \bullet : \quad & f(\theta) \bullet \mathbf{u} = \left(\sum_{j=0}^{m} c_j \theta^j \right) \bullet \mathbf{u} \mapsto f(\varphi)(\mathbf{u}) = \sum_{j=0}^{m} c_j \varphi^j(\mathbf{u}), \end{aligned} \tag{3.6}$$

where $c_j \in \mathcal{F}$ and $\mathbf{u} \in \mathcal{V}$. This action satisfies, by definition, the composition property $(fg) \bullet \mathbf{u} = f \bullet (g \bullet \mathbf{u})$ for all $f, g \in \mathcal{S}$ and $\mathbf{u} \in \mathcal{V}$. The dot $\bullet$ is sometimes omitted for brevity when the context is clear.

With this framework, the set of solutions V_f in $\mathcal{V}$ to an Ore polynomial equa-

tion $f(\theta) \bullet \mathbf{u} = 0$ (or $f(\varphi)(\mathbf{u}) = 0$) is:

$$V_f := \{\mathbf{u} \in \mathcal{V} \mid f(\varphi)(\mathbf{u}) = 0\}, \tag{3.7}$$

where φ is the specific $\mathcal{F}$-pseudo-linear map defining the action of θ on $\mathcal{V}$. The set V_f forms a vector space over $\mathrm{Const}(\mathcal{F})$, provided $\mathcal{V}$ is an $\mathcal{F}$-vector space.

Consider an element $\mathbf{u} \in V_f$, i.e. $f(\varphi)(\mathbf{u}) = 0$. The set $\mathcal{S}\,f = \{sf \mid s \in \mathcal{S}\}$ is the principal left ideal generated by f in $\mathcal{S}$. The map

$$\begin{aligned} \Phi \;:\; & \mathcal{S}\,/\,\mathcal{S}\,f \to \mathcal{V} \\ & s + \mathcal{S}\,f \mapsto s \bullet \mathbf{u} \end{aligned} \tag{3.8}$$

is a well-defined homomorphism of left $\mathcal{S}$-modules. The element $\Phi(1_{\mathcal{S}} + \mathcal{S}\,f) = 1_{\mathcal{S}} \bullet \mathbf{u} = \mathbf{u}$ (where $1_{\mathcal{S}}$ is the multiplicative identity in $\mathcal{S}$) can be regarded as a *generic solution* associated with the factor module $\mathcal{S}\,/\,\mathcal{S}\,f$, in the sense that any other element $s \bullet \mathbf{u}$ (derived from $s + \mathcal{S}\,f \in \mathcal{S}\,/\,\mathcal{S}\,f$) is an image under Φ. This role as a generic solution is well-founded, since $\Phi(1_{\mathcal{S}} + \mathcal{S}\,f)$ evaluates to $1_{\mathcal{S}} \bullet \mathbf{u} = \mathbf{u}$, and $\mathbf{u}$ was chosen such that $f(\varphi)(\mathbf{u}) = 0$.

Remark 3.3.7. *If $\mathcal{F}$ is viewed as a left module over itself, setting $\varphi = \delta$ in the definition of an $\mathcal{F}$-pseudo-linear map (Definition 3.2.2) demonstrates that δ itself acts as such a map; indeed, this substitution directly yields the σ-derivation rule (Equation 3.1). Similarly, if $\delta = 0$, setting $\varphi = \sigma$ shows that the endomorphism σ also satisfies the $\mathcal{F}$-pseudo-linear map condition (Equation 3.2), a case of particular interest in this study. Thus, the specific $\mathcal{F}$-pseudo-linear action φ of the indeterminate θ simplifies in certain ring structures:*

$$\varphi = \begin{cases} \sigma & \text{if } \delta = 0 \text{ (characterising the action } \varphi \text{ of } \theta \text{ in } \mathcal{F}[\theta; \sigma, 0]), \\ \delta & \text{if } \sigma = \mathrm{id} \text{ (characterising the action } \varphi \text{ of } \theta \text{ in } \mathcal{F}[\theta; \mathrm{id}, \delta]). \end{cases} \tag{3.9}$$

In the general Ore ring $\mathcal{F}[\theta; \sigma, \delta]$, where σ may not be the identity and δ may be

non-zero, the action φ *of* θ *is a more complex map whose structure is defined by both* σ *and* δ *through the pseudo-linear condition* $\varphi(ab) = \sigma(a)\varphi(b) + \delta(a)b$ *(when* φ *acts on* $\mathcal{F}$*).*

3.3.1 Classification of Ore Rings over Commutative Fields

It is known that if the base ring $\mathcal{F}$ is a commutative field, any univariate Ore polynomial ring $\mathcal{F}[\theta; \sigma, \delta]$ can be simplified (or transformed) as shown in the following lemma (see for example, [15, Lemma 1] or [32, p. 498, Proposition 3.1]).

Lemma 3.3.8. *Let* $\mathcal{F}$ *be a commutative field, let* σ *be an injective endomorphism of* $\mathcal{F}$*, and let* δ *be a* σ*-derivation of* $\mathcal{F}$*.*

1. ***Automorphism Case:*** *If* $\sigma \neq \mathrm{id}_{\mathcal{F}}$*, then there exists an element* $\alpha \in \mathcal{F}$ *such that* $\delta = \alpha(\sigma - \mathrm{id}_{\mathcal{F}})$*, which means the ring* $\mathcal{F}[\theta; \sigma, \delta]$ *is isomorphic to a pure automorphism-type ring.*

2. ***Differential Case:*** *If* $\sigma = \mathrm{id}_{\mathcal{F}}$*, then* δ *is a standard derivation on* $\mathcal{F}$ *(as* $\delta(ab) = a\delta(b) + \delta(a)b$*), and the ring* $\mathcal{F}[\theta; \mathrm{id}_{\mathcal{F}}, \delta]$ *is a ring of differential operators.*

The possible options in the lemma above demonstrate that, over a commutative field $\mathcal{F}$, any Ore polynomial ring $\mathcal{F}[\theta; \sigma, \delta]$ falls into one of three categories:

1. The *commutative case* (when $\sigma = \mathrm{id}_{\mathcal{F}}$ and $\delta = 0$).

2. The *automorphism type* (or difference/shift type, when $\sigma \neq \mathrm{id}_{\mathcal{F}}$; in this case, δ is related to σ and can be eliminated via a change of variable).

3. The *differential type* (when $\sigma = \mathrm{id}_{\mathcal{F}}$ and $\delta \neq 0$).

Operator Evaluation

One of the main differences encountered when dealing with evaluations in non-commutative rings is that evaluation maps behave quite differently compared to

the commutative case (where values are simply substituted). This is principally because non-commutative evaluation maps are not compatible with multiplication in the same straightforward manner.

The aim is usually to find a suitable evaluation map that is not only a ring homomorphism but also correctly handles the interactions between indeterminates, especially in multivariate settings where indeterminates are typically assumed to commute with each other. For example, consider the ring $\mathbb{C}[\theta_1;\sigma_1][\theta_2;\sigma_2]$ over the complex numbers $\mathbb{C}$. If one were to substitute i for θ_1 in a multivariate expression such as $\theta_1\theta_2$, the result would be $i\theta_2$. However, substituting i for θ_1 in the commuting expression $\theta_2\theta_1$ would lead to $\theta_2 i$. If σ_2 (the automorphism associated with θ_2 acting on coefficients from $\mathbb{C}[\theta_1;\sigma_1]$) is non-trivial on i (e.g., if i is a coefficient subject to σ_2), then $\theta_2 i$ becomes $\sigma_2(i)\theta_2$. Thus, the results $i\theta_2$ and $\sigma_2(i)\theta_2$ may not be identical (depending on the definition of σ_2), illustrating a challenge in defining consistent evaluation.

Although the fundamental concept of evaluation (assigning a value or operator to an indeterminate) persists, various non-commutative evaluation maps serve different purposes. Beyond explicit evaluation formulae, concepts include evaluation via the remainder theorem [32], the product formula for evaluations [86], and the specific operator evaluation discussed here [10], among others.

A key feature of Ore polynomials is that their elements can be interpreted as operators. This operator-focused view provides a natural motivation for exploring evaluation maps where the indeterminate itself is substituted with an appropriate operator. A particularly interesting target for such an evaluation is the map σ itself, especially when $\delta = 0$. Since σ is an endomorphism (often an automorphism in the contexts considered), evaluating the indeterminate θ at σ can lead to a well-behaved ring homomorphism. This approach is known as *operator evaluation* [10].

To formalise this, one defines the *ring of σ-operators*[1] [10]:

$$\mathcal{F}_{\circ}[\sigma] = \left\{ \sum_{j=0}^{m} c_j \sigma^j \mid c_j \in \mathcal{F} \right\}. \tag{3.10}$$

The set of polynomials in an indeterminate σ over a field $\mathcal{F}$ can form a non-commutative ring, which is an instance of an Ore ring of operators of the type $\mathcal{F}_{\circ}[\sigma; \sigma, 0]$ with multiplication defined by composition. Here, the endomorphism is σ itself, and the associated derivation is the zero map.

This structure is motivated by viewing polynomials as operators, where multiplication is defined by operator composition ($\circ$). For any $c \in \mathcal{F}$, let L_c be the left multiplication operator. The interaction between the map σ and L_c is governed by the operator equality $\sigma \circ L_c = L_{\sigma(c)} \circ \sigma$. This identity leads to the fundamental commutation rule in the abstract ring:

$$\sigma c = \sigma(c)\sigma, \quad \forall c \in \mathcal{F}. \tag{3.11}$$

For any two polynomials $f = \sum_{j=0}^{m} c_j \sigma^j$ and $g = \sum_{l=0}^{k} d_l \sigma^l$, the ring operations are as follows:

Addition: Addition is performed component-wise:

$$f + g = \sum_{p=0}^{\max(m,k)} (c_p + d_p)\sigma^p. \tag{3.12}$$

Multiplication: Multiplication is determined by extending the rule (3.11) via the distributive law. The product is given by:

$$f \cdot g = \left(\sum_{j=0}^{m} c_j \sigma^j \right) \left(\sum_{l=0}^{k} d_l \sigma^l \right) = \sum_{j=0}^{m} \sum_{l=0}^{k} c_j \sigma^j (d_l) \sigma^{j+l}. \tag{3.13}$$

[1] This structure is also referred to, in literature, as a "skew polynomial ring in σ", a "difference ring", or "the ring of difference operators" when σ is a shift operator.

The notation $\mathcal{F}[\sigma; \circ]$, as used in the original text, effectively captures this interpretation of multiplication as operator composition.

Operator Representation and Evaluation ($\delta = 0$)

For the specific case of Ore rings with zero derivation $\mathcal{F}[\theta; \sigma, 0]$, polynomials can be interpreted as linear operators on $\mathcal{F}$. This is formalized by a ring homomorphism.

Lemma 3.3.9 (Operator Homomorphism, cf. [10]). *Consider the Ore ring $\mathcal{F}[\theta; \sigma, 0]$. The map $\mathrm{eval}_{(\theta\text{-}\sigma)}$ that replaces the indeterminate θ with the endomorphism σ;*

$$\mathrm{eval}_{(\theta\text{-}\sigma)}: \begin{array}{rcl} \mathcal{F}[\theta; \sigma, 0] & \to & \mathcal{F}_{\circ}[\sigma; \sigma, 0] \\ f(\theta) = \sum_{j=0}^{m} c_j \theta^j & \mapsto & f(\sigma) := \sum_{j=0}^{m} c_j \sigma^j \end{array}$$

is a homomorphism of rings.

Building on this, the evaluation of such an operator $f(\sigma)$ at an element $a \in \mathcal{F}$ is defined as follows:

Definition 3.3.10 (Operator Evaluation at a Point [10]). *Let $f(\theta) = \sum_{j=0}^{m} c_j \theta^j \in \mathcal{F}[\theta; \sigma, 0]$. The operator evaluation of $f(\theta)$ (via $\theta \mapsto \sigma$) at an element $a \in \mathcal{F}$ is defined as:*

$$\mathrm{eval}_{(\theta\text{-}\sigma)(a)}: \begin{array}{rcl} \mathcal{F}[\theta; \sigma, 0] & \to & \mathcal{F} \\ f(\theta) = \sum_{j=0}^{m} c_j \theta^j & \mapsto & f(\sigma)(a) := \sum_{j=0}^{m} c_j \sigma^j(a) \end{array}$$

If $f(\sigma)(a) = 0$ then $\mathbf{u} = a$ is called a solution of the operator equation $f(\sigma)(\mathbf{u}) = 0$ (where $\mathbf{u}$ is a placeholder for the argument). When the context of evaluating $f(\theta)$ via $\theta \mapsto \sigma$ and subsequently applying it to a is clear, the value $f(\sigma)(a)$ is sometimes denoted by $f^*(a)$.

Polynomial Evaluation and Product Rule (General Case)

For the general Ore ring $\mathcal{F}[\theta;\sigma,\delta]$, evaluation is defined via polynomial right division, a concept distinct from the operator evaluation discussed previously. The following definition and lemmas, which formalise this standard approach, are well-established in the literature.

Lemma 3.3.11 (Remainder Theorem [32]). *For any $f(\theta) \in \mathcal{F}[\theta;\sigma,\delta]$ and $a \in \mathcal{F}$, there exists a unique polynomial $q(\theta)$ and a constant $r \in \mathcal{F}$ such that $f(\theta) = q(\theta)\cdot(\theta - a) + r$. This constant r is the* right evaluation *of f at a, denoted $f(a)$.*

This evaluation method has a non-trivial product rule that reflects the non-commutative nature of the ring. To formulate this rule, the concept of conjugacy is essential and is therefore introduced first, followed by the formal statement of the product rule theorem.

Definition 3.3.12 ((σ,δ)-Conjugacy). *An element $b \in \mathcal{F}$ is a* (σ,δ)-conjugate *of $a \in \mathcal{F}$ if there exists some $c \in \mathcal{F}^\times$ such that*

$$b = a^c := \sigma(c)ac^{-1} + \delta(c)c^{-1}.$$

Conjugacy is an equivalence relation, and the set of all conjugates of a forms its conjugacy class $\mathrm{Cl}(a)$.

Lemma 3.3.13 (Product Rule for Evaluation [85]). *Let $f(\theta), g(\theta) \in \mathcal{F}[\theta;\sigma,\delta]$ and $a \in \mathcal{F}$. The evaluation of the product $h(\theta) = f(\theta)g(\theta)$ at a is given by:*

$$(fg)(a) = \begin{cases} 0 & \text{if } g(a) = 0 \\ f\left(a^{g(a)}\right)\cdot g(a) & \text{if } g(a) \neq 0 \end{cases} \tag{3.14}$$

where $a^{g(a)}$ is the (σ,δ)-conjugate of a by the non-zero element $g(a) \in \mathcal{F}$.

3.4 Euclidean Domain, Ore Condition, and Related Concepts

This section reviews the definition of a Euclidean domain, crucial for algorithmic polynomial manipulation, and discusses the Ore condition, which is essential for constructing fraction fields and defining certain operations in non-commutative rings.

Definition 3.4.1 (Right Euclidean Domain)**.** *A (not necessarily commutative) domain $\mathcal{R}$, endowed with a map (norm) $N : \mathcal{R} \setminus \{0\} \to \mathbb{N}_0$, is a* right Euclidean domain *with respect to N if for all $f, g \in \mathcal{R}$ with $g \neq 0$:*

(i) There exist $q, r \in \mathcal{R}$ such that $f = qg + r$, where $r = 0$ or $N(r) < N(g)$.

(ii) $N(f) \leq N(fg)$ for $f \neq 0$. (This ensures N respects multiplication reasonably).

The elements q and r are termed the right quotient $(\mathrm{rquo}(f, g))$ *and right remainder* $(\mathrm{rrem}(f, g))$ *resulting from the right division of f by g. A ring $\mathcal{R}$ in which these elements are uniquely determined is called a unique right Euclidean domain.*

The skew polynomial ring $\mathcal{S} = \mathcal{F}[\theta; \sigma, \delta]$ (where $\mathcal{F}$ is a skew field and σ an automorphism) is a right (and left) Euclidean domain with $N = \deg$. The quotient and remainder are unique. Algorithm 1 details right division [17, Algorithm 2, P. 39].

Algorithm 1: Ore Polynomial Right Division

Input : $f, g \in \mathcal{F}[\theta; \sigma, \delta]$, $g \neq 0$. ($\mathcal{F}$ is a skew field, σ an automorphism).

Output: $q = \text{rquo}(f, g)$, $r = \text{rrem}(f, g)$ such that $f = qg + r$ and ($r = 0$ or $\deg(r) < \deg(g)$).

1 $r := f$; $q := 0$

2 $m := \deg(g)$

3 $b_m := \text{lc}(g)$

4 **while** $r \neq 0$ **and** $\deg(r) \geq m$ **repeat**

5 $c_{\deg(r)} := \text{lc}(r)$

6 $s := c_{\deg(r)}(\sigma^{\deg(r)-m}(b_m))^{-1}\theta^{\deg(r)-m}$ // b_m^{-1} must be σ-applied

7 $q := q + s$

8 $r := r - sg$

9 **return** (q, r)

However, multivariate skew polynomial rings such as $\mathcal{F}[\theta_1; \sigma_1, \delta_1][\theta_2; \sigma_2, \delta_2]$ are not generally Euclidean domains with respect to, for example $\deg_{\theta_2}$, as the coefficients are themselves polynomials in θ_1 and may not be invertible.

Consequently, the Euclidean division algorithm with respect to θ_2 is not always applicable, meaning unique quotients and remainders satisfying the degree condition might not exist. The following assertion illustrates this.

Assertion. Let $\mathcal{F}$ be a (skew) field, and consider the bivariate Ore polynomial ring

$$\mathcal{R} \;=\; \mathcal{F}[\theta_1; \sigma_1, \delta_1]\,[\theta_2; \sigma_2, \delta_2].$$

It is asserted that $\mathcal{R}$ is not generally a Euclidean domain when viewed as polynomials in one indeterminate (e.g., θ_2) with coefficients from the ring of polynomials in the other indeterminate (i.e. $\mathcal{R} \;=\; \mathcal{S}_1\,[\theta_2; \sigma_2, \delta_2]$ where $\mathcal{S}_1 = \mathcal{F}[\theta_1; \sigma_1, \delta_1]$), using the degree in the chosen indeterminate (e.g. $\deg_{\theta_2}$) as the Euclidean norm.

Proof. Suppose, for contradiction, that $\mathcal{R}$ were a (right) Euclidean domain with respect to the norm $N = \deg_{\theta_2}$. Then for any $f, g \in \mathcal{R}$ with $g \neq 0$, there must

exist $q, r \in \mathcal{R}$ such that

$$f \;=\; q\,g \;+\; r,$$

where either $r = 0$ or $N(r) = \deg_{\theta_2}(r) < \deg_{\theta_2}(g)$.

Consider the specific case of $f = 1_{\mathcal{R}}$ (the identity in $\mathcal{R}$, which is $1_{\mathcal{F}}$) and $g = \theta_1$. Both $1_{\mathcal{R}}$ and θ_1 are elements of $\mathcal{S}_1 = \mathcal{F}[\theta_1; \sigma_1, \delta_1]$, and thus are in $\mathcal{R}$. As polynomials in θ_2:

- $f = 1_{\mathcal{R}}$ has $\deg_{\theta_2}(1_{\mathcal{R}}) = 0$.
- $g = \theta_1$ has $\deg_{\theta_2}(\theta_1) = 0$.

The Euclidean condition on the remainder r requires $\deg_{\theta_2}(r) < \deg_{\theta_2}(\theta_1) = 0$. This implies that r must be the zero polynomial, $r = 0$, since only the zero polynomial has a degree less than 0 (conventionally, $\deg(0) = -\infty$).

The division equation thus simplifies to $1_{\mathcal{R}} = q\,\theta_1$. For this equality to hold, the element $q \in \mathcal{R} = \mathcal{S}_1[\theta_2; \sigma_2, \delta_2]$ must have $\deg_{\theta_2}(q\theta_1) = \deg_{\theta_2}(1_{\mathcal{R}}) = 0$. Since $\theta_1 \in \mathcal{S}_1$ has degree 0 in θ_2, this implies $\deg_{\theta_2}(q) = 0$. Therefore, q must be an element of $\mathcal{S}_1$, say $q_0(\theta_1) \in \mathcal{F}[\theta_1; \sigma_1, \delta_1]$. The equation becomes $1_{\mathcal{F}} = q_0(\theta_1) \cdot \theta_1$. This would mean that $q_0(\theta_1)$ is a left inverse of θ_1 within the ring $\mathcal{F}[\theta_1; \sigma_1, \delta_1]$.

However, θ_1 is an indeterminate over the (skew) field $\mathcal{F}$ and is therefore not invertible in the polynomial ring $\mathcal{F}[\theta_1; \sigma_1, \delta_1]$ (unless this ring degenerates to $\mathcal{F}$ itself, which is not the case if θ_1 is a true indeterminate). Consequently, no such $q_0(\theta_1)$ exists in $\mathcal{F}[\theta_1; \sigma_1, \delta_1]$, and thus no such q exists in $\mathcal{R}$.

This contradicts the initial assumption that Euclidean division $1_{\mathcal{R}} = q\theta_1 + r$ with the required remainder condition could be performed. Therefore, $\mathcal{R}$ is not generally a Euclidean domain. □

Remark 3.4.2. *This conclusion aligns with a more general algebraic fact: any Euclidean domain must be a principal ideal domain (PID). However, multivariate polynomial rings such as $\mathcal{F}[\theta_1, \theta_2]$ (even when commutative, i.e. $\sigma_i = \mathrm{id}, \delta_i = 0$) are not PIDs. For instance, the ideal $\langle \theta_1, \theta_2 \rangle$ generated by θ_1 and θ_2 is not principal. This structural property provides another reason why such iterated Ore*

polynomial rings (which are multivariate in nature) generally fail to be Euclidean domains.

While not always Euclidean, multivariate Ore rings can satisfy the Ore condition:

Proposition 3.4.3 (Ore Condition, cf. [111]). *A ring $\mathcal{R}$ is said to have the property of* left Ore condition *if for all non-zero $a, b \in \mathcal{R}$, the intersection of principal left ideals ${}_{\mathcal{R}}\langle a\rangle \cap {}_{\mathcal{R}}\langle b\rangle \neq \{0\}$. Equivalently, there exist non-zero $d_1, d_2 \in \mathcal{R}$ such that $d_1 a = d_2 b$. If a domain fulfills both the left and right Ore conditions, it is known as an* Ore domain.

Remark 3.4.4 (Embedding in a Skew Field of Fractions). *The Ore condition is necessary and sufficient for an Ore domain $\mathcal{R}$ to be embeddable into a skew field of fractions [110]. For $\mathcal{S} = \mathcal{F}[\theta; \sigma, \delta]$ where $\mathcal{F}$ is a skew field and σ an automorphism, $\mathcal{S}$ is a PID, hence an Ore domain, and can be embedded in $\mathcal{F}(\theta; \sigma, \delta)$. This is essential for operations such as matrix inversion for Dieudonné determinants (an operation detailed in subsequent sections). Iterated Ore extensions over a field also form Ore domains. For example, if $\mathcal{R}$ is a left Ore ring, then $\mathcal{R}[\theta; \sigma, \delta]$ is also a left Ore ring provided σ is injective [33, Propostion 1.1.4].*

In Euclidean domains such as $\mathcal{F}[\theta; \sigma, \delta]$, the extended Euclidean algorithm computes common divisors and Bezout cofactors.

Definition 3.4.5 (Common Divisors and Multiples). *Let f, g be elements in a ring $\mathcal{R}$.*

- *A* greatest common right divisor $(\mathrm{gcrd}(f, g))$ *is a common right divisor d of f and g such that any other common right divisor of f and g is also a right divisor of d. It is unique up to left multiplication by a unit.*

- *A* least common left multiple $(\mathrm{lclm}(f, g))$ *is a common left multiple m of f and g such that any other common left multiple of f and g is also a left multiple of m. It is unique up to right multiplication by a unit. The ideal $\mathcal{R}f \cap \mathcal{R}g = \mathcal{R}(\mathrm{lclm}(f, g))$.*

Analogous definitions hold for greatest common left divisor ($\mathrm{gcld}(f,g)$) *and least common right multiple* ($\mathrm{lcrm}(f,g)$). *In a PID such as* $\mathcal{F}[\theta;\sigma,\delta]$, *GCRDs and LCLMs always exist.*

Algorithm 2 computes the $\mathrm{gcrd}(f,g)$ and cofactors s,t such that $sf+tg=\mathrm{gcrd}(f,g)$. It can be adapted to find u,v such that $uf=vg=\mathrm{lclm}(f,g)$.

Algorithm 2: Skew Extended Euclidean Algorithm

Input : $f,g\in\mathcal{F}[\theta;\sigma,\delta]$, where $\mathcal{F}$ is a (skew) field and σ an automorphism.

Output: $d\in\mathcal{F}[\theta;\sigma,\delta]$, where d is a gcrd of f and g; $s,t\in\mathcal{F}[\theta;\sigma,\delta]$ such that $sf+tg=d$.

1 $r_0:=f$; $s_0:=1_{\mathcal{F}}$; $t_0:=0$

2 $r_1:=g$; $s_1:=0$; $t_1:=1_{\mathcal{F}}$

3 $i:=1$

4 **while** $r_i\neq 0$ **repeat**

5 $(q_{i+1},r_{i+1}):=(\mathrm{rquo}(r_{i-1},r_i),\mathrm{rrem}(r_{i-1},r_i))$ // Using Algorithm 1

6 $s_{i+1}:=s_{i-1}-s_iq_{i+1}$

7 $t_{i+1}:=t_{i-1}-t_iq_{i+1}$

8 $i:=i+1$

9 $d:=r_{i-1}$; $s:=s_{i-1}$; $t:=t_{i-1}$ // Normalize d,s,t if d is not monic, e.g. $u:=\mathrm{lc}(d)^{-1}$, $d:=ud, s:=us, t:=ut$

10 **return** (d,s,t)

3.5 Similarity and Unique Factorisation in Non-Commutative Rings

In commutative algebra, a principal ideal domain (PID) is also a unique factorisation domain (UFD), where any non-zero non-unit element can be factored into a product of irreducible elements, unique up to reordering and multiplication by units (associates). While Ore polynomial rings such as $\mathcal{F}[\theta;\sigma,\delta]$ (where $\mathcal{F}$ is a

skew field and σ is an automorphism) are non-commutative PIDs, they are not UFDs in this classical sense. For example, a polynomial like $\theta a\theta + \theta$ might have distinct factorisations, such as $\theta(a\theta + 1)$ and $(\theta a + 1)\theta$, whose factors are not merely unit multiples of each other.

To correctly generalise the concept of unique factorisation to such non-commutative settings, the notion of "associates" is replaced by the broader concept of "similarity".

Definition 3.5.1 (Similarity, cf. [35, 73])**.** *Let $\mathcal{R}$ be a domain. Two elements $f, g \in \mathcal{R}$ are said to be* similar*, denoted $f \sim g$, if the left $\mathcal{R}$-modules $\mathcal{R}/_{\mathcal{R}}\langle f\rangle$ and $\mathcal{R}/_{\mathcal{R}}\langle g\rangle$ are isomorphic. In a PID such as $\mathcal{F}[\theta;\sigma,\delta]$, a simpler characterisation is that $f \sim g$ if and only if there exists an element $u \in \mathcal{R}$ such that $\mathcal{R} = \mathcal{R}f + \mathcal{R}u$ and their least common left multiple is* $\operatorname{lclm}(f, u) = gu$.

An element $p \in \mathcal{R}$ is *irreducible* (or an atom) if it is not a unit and any factorisation $p = ab$ implies that either a or b must be a unit. With these concepts, unique factorisation can be defined for non-commutative rings.

Definition 3.5.2 (Non-commutative Unique Factorisation Domain, cf. [35, 53])**.** *A domain $\mathcal{R}$ is called a* unique factorisation domain in the non-commutative sense *(or a UFD$_{\approx}$) if every non-zero non-unit element can be factored into a product of irreducible elements, and this factorisation is unique up to the order of factors and the similarity of corresponding factors. That is, if*

$$p_1 p_2 \cdots p_n = q_1 q_2 \cdots q_m$$

are two factorisations into irreducibles, then $n = m$, and there exists a permutation π of $\{1, \ldots, n\}$ such that p_i is similar to $q_{\pi(i)}$ for each $i = 1, \ldots, n$.

This definition provides the necessary non-commutative counterpart to the classical UFD establishing a fundamental property central to the results of this research.

Lemma 3.5.3 (see for example, [111] or [35]). *Every principal ideal domain (PID), including non-commutative PIDs such as the Ore polynomial ring $\mathcal{F}[\theta;\sigma,\delta]$ (where $\mathcal{F}$ is a skew field and σ is an automorphism), is a unique factorisation domain in this non-commutative sense (UFD$_{\approx}$).*

Remark 3.5.4. *Lemma 3.5.3 thus confirms that factorisation into irreducibles in Ore polynomial rings like $\mathcal{F}[\theta;\sigma,\delta]$ is well-defined and unique up to the similarity and reordering of factors, as specified in Definition 3.5.2.*

3.6 Dieudonné Determinant

Extending the concept of resultants to non-commutative settings, particularly for matrices whose entries are Ore polynomials, requires a generalisation of the determinant. The Dieudonné determinant, introduced by Jean Dieudonné in 1943 [44], serves this purpose for matrices over skew fields. It provides a homomorphism from the general linear group over a skew field to an abelian group.

Definition 3.6.1 (Invertible Matrix). *Let $\mathcal{D}$ be a skew field. A square matrix $M \in M_k(\mathcal{D})$ (the ring of $k \times k$ matrices over $\mathcal{D}$) is* invertible *if there exists $M^{-1} \in M_k(\mathcal{D})$ such that $MM^{-1} = M^{-1}M = I_k$, where I_k is the $k \times k$ identity matrix. The group of such invertible matrices is the general linear group* $\mathrm{GL}_k(\mathcal{D})$.

It is known (cf. [34, p. 349] or [47, §19, Theorem 1] also see [74]) that any invertible matrix $M \in \mathrm{GL}_k(\mathcal{D})$ can be decomposed, for instance in the form of $M = L'DU'P$, where L' and U' are unitriangular matrices (triangular with ones on the diagonal), D is a diagonal matrix $D = \mathrm{diag}(d_1, \ldots, d_k)$, and P is a permutation matrix. The diagonal entries d_i are unique up to their order and multiplication by elements from the commutator subgroup $[\mathcal{D}^\times, \mathcal{D}^\times]$. This (or related decompositions such as Bruhat normal form) provides a basis for defining the determinant.

Definition 3.6.2 (Dieudonné Determinant). *Let $\mathcal{D}$ be a (skew) field. Let $\mathcal{D}^\times$ be its multiplicative group and $[\mathcal{D}^\times, \mathcal{D}^\times]$ be the commutator subgroup of $\mathcal{D}^\times$*

(which is a normal subgroup, c.f. normalizer of a subgroup [126, Exercise 1.11.3]). The Dieudonné determinant, denoted by Det*, is a group homomorphism* $\mathrm{Det} : \mathrm{GL}_k(\mathcal{D}) \to \mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$*, where* k *is the matrix dimension* $(k \times k)$*. For an invertible matrix* $M \in \mathrm{GL}_k(\mathcal{D})$*, if* M *is brought to a diagonal form* $D = \mathrm{diag}(d_1, \ldots, d_k)$ *through operations that preserve the determinant class (e.g. as in the decomposition* $M = L'DU'P$*), its determinant is defined as:*

$$\mathrm{Det}(M) = [\mathrm{sgn}(P) \cdot d_1 d_2 \cdots d_k], \tag{3.15}$$

where $\mathrm{sgn}(P)$ *is the sign* (± 1) *of the permutation* P *involved in the decomposition, and* $[x]$ *denotes the class of* $x \in \mathcal{D}^\times$ *in the abelian quotient group* $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$*. If* $M \notin \mathrm{GL}_k(\mathcal{D})$ *(i.e.* M *is singular), then* $\mathrm{Det}(M) = 0$*. The property* $[fg] = [f][g] = [g][f]$ *holds in* $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$*.*

It is a known property that for any invertible matrix M and any signed permutation matrix P_s (a permutation matrix where rows might be multiplied by -1 such that $\mathrm{Det}(P_s) = [1]$), $\mathrm{Det}(P_s M) = \mathrm{Det}(M)$ [34, p. 352]. By using appropriate signed permutations in the decomposition, or by noting that the sign often gets incorporated into the choice of representative from the commutator class for specific applications (such as resultants where only vanishing matters), the formula is usually simplified to:

$$\mathrm{Det}(M) = [d_1 d_2 \cdots d_k]. \tag{3.16}$$

One of the fundamental properties of the Dieudonné determinant is that its value (as a class in $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$) is invariant under elementary row operations of adding a multiple of one row to another, and it behaves predictably under row swaps and row scaling, analogous to determinants in the commutative case [1].

Remark 3.6.3. *The definition of the Dieudonné determinant (Definition 3.6.2) is given via a diagonal form. However, it is usually computed by transforming the matrix* M *into an upper (or lower) triangular form (say* T*) using elementary row*

operations. If t_{ii} are the diagonal elements of T (which are guaranteed to be non-zero if M is invertible), then $\mathrm{Det}(M) = [\mathrm{sgn}(\pi) \cdot \prod t_{ii}]$, *where* $\mathrm{sgn}(\pi)$ *accounts for row swaps. As with Equation 3.16, this is simplified to* $\mathrm{Det}(M) = [\prod t_{ii}]$. *This approach is particularly useful when working with matrices whose entries are from a polynomial ring (e.g. an Ore polynomial ring $\mathcal{R}[\theta; \sigma, \delta]$), as triangularisation can often be performed while keeping entries within that original ring, rather than requiring computation in the full skew field of fractions for a diagonal form. This aspect is relevant for computing resultants as polynomials and is further discussed in Lemma 3.6.7.*

Let $\mathcal{D}^\times$ be the multiplicative group of non-zero elements of $\mathcal{D}$. Let $[\mathcal{D}^\times, \mathcal{D}^\times]$ be its commutator subgroup, generated by elements of the form $aba^{-1}b^{-1}$ for $a, b \in \mathcal{D}^\times$. The quotient group $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$ is abelian. The Dieudonné determinant maps $M_k(\mathcal{D})$ to $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times] \cup \{0\}$.

An equivalent definition of the Dieudonné determinant is as follows.

Definition 3.6.4 (Dieudonné Determinant, cf. [1, 44])**.** *Let $M \in M_k(\mathcal{D})$. The Dieudonné determinant* $\mathrm{Det}(M)$ *is defined as follows:*

1. *If M is not invertible (i.e. $M \notin \mathrm{GL}_k(\mathcal{D})$), then* $\mathrm{Det}(M) = 0$.
2. *If $M \in \mathrm{GL}_k(\mathcal{D})$, then M can be transformed into a diagonal matrix $D = \mathrm{diag}(d_1, \ldots, d_{k-1}, d'_k)$ by elementary row operations of adding a multiple of one row to another, and potentially a final permutation. More standardly, M can be reduced to an upper (or lower) triangular matrix T using only row operations of type (i) (swapping two rows) and type (iii) (adding a left multiple of one row to another). If $t_{11}, \ldots, t_{kk}$ are the diagonal entries of such a T (all non-zero as M is invertible), then*

$$\mathrm{Det}(M) = [t_{11} \cdots t_{kk}] \pmod{[\mathcal{D}^\times, \mathcal{D}^\times]},$$

where p is the number of row swaps used.

This determinant is well-defined (independent of the specific sequence of row operations) and is multiplicative: $\mathrm{Det}(AB) = \mathrm{Det}(A)\,\mathrm{Det}(B)$ *for* $A, B \in M_k(\mathcal{D})$. *If* M *is diagonal with entries* $d_1, \ldots, d_k$, $\mathrm{Det}(M) = [d_1 \cdots d_k]$.

3.6.1 Elementary Operations and Properties of the Dieudonné Determinant

This subsection reviews elementary row operations in the non-commutative context and discusses key properties of the Dieudonné determinant, particularly its well-definedness modulo commutators. These concepts are fundamental to computing the determinant and understanding its behaviour. For further details on elementary operations and matrix theory over skew fields, see for example, [34, §9.2] or [1, Chapter IV].

Let $\mathcal{D}$ be a (skew) field. Recall $M_n(\mathcal{D})$ denotes the ring of all $n \times n$ matrices over $\mathcal{D}$. The following typical elementary row operations are compatible with the Dieudonné determinant, with their effects on the determinant class noted:

- **Type (i):** Swapping two rows, $R_i \leftrightarrow R_j$. This operation, corresponding to left multiplication by a permutation matrix P_{ij}, multiplies the Dieudonné determinant by $[-1]$.
- **Type (ii):** Multiplying a row R_i by an invertible scalar $u \in \mathcal{D}^\times$ (from the left). This corresponds to left multiplication by $E_i(u) = I_n + (u-1)e_{ii}$ (where e_{ii} is a matrix unit) and multiplies the Dieudonné determinant by $[u]$.
- **Type (iii):** Adding q times row j to row i ($R_i \leftarrow R_i + qR_j$, for $i \neq j, q \in \mathcal{D}$). This corresponds to left multiplication by an elementary matrix $E_{ij}(q) = I_n + qe_{ij}$ and leaves the Dieudonné determinant unchanged (i.e. multiplies by $[1]$).

Similarly to the commutative case, these operations are employed to transform a matrix into a simpler (e.g. triangular or diagonal) form from which the Dieudonné determinant can be computed easily.

As noted, for computational convenience, it is preferable to use *signed permutations*, which are constructed to ensure their Dieudonné determinant is [1]. A signed permutation can be constructed entirely from type (iii) operations, which do not alter the determinant. For instance, applying the sequence $E_{ij}(1)E_{ji}(-1)E_{ij}(1)$ to a 2×2 identity matrix I_2 results in the matrix $\left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right)$, which permutes rows (with a sign change) but still has a Dieudonné determinant of [1]. This allows permutations to be implemented within a triangularisation process in a *determinant-preserving* manner (modulo commutators).

A fundamental property of Dieudonné determinant is its uniqueness modulo commutators.

Theorem 3.6.5. *The Dieudonné determinant* $\mathrm{Det}(M)$ *of a matrix* $M \in M_n(\mathcal{D})$ *yields a unique element in the quotient group* $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$ *if* M *is invertible, or* 0 *if* M *is singular.*

Proof. The proof can proceed by induction on n, the size of the matrix. For $n = 1$, $M = (x)$. $\mathrm{Det}(M) = [x]$ if $x \in \mathcal{D}^\times$, and 0 if $x = 0$. This is unique. Assume the theorem holds for matrices of size $(n-1) \times (n-1)$. For an $n \times n$ matrix M: if M is singular, $\mathrm{Det}(M) = 0$. If M is invertible, its first column must be non-zero. By row swaps (if necessary), ensure the $(1,1)$ entry a_{11}, is non-zero. Multiply row 1 by a_{11}^{-1} to make the $(1,1)$ entry 1. This scales the determinant by $[a_{11}^{-1}]$. Then, for $j > 1$, add $-a_{j1}$ times the new row 1 to row j to make all other entries in the first column zero. These latter operations do not change the determinant class. The matrix becomes:

$$\begin{pmatrix} 1 & \text{row vector} \\ \mathbf{0} & M' \end{pmatrix}$$

where M' is an $(n-1) \times (n-1)$ invertible matrix. Then $\mathrm{Det}(M)$ (after accounting for the initial scaling by $[a_{11}]$ from the inverse operation) is given by $\mathrm{Det}(M')$. By the induction hypothesis, $\mathrm{Det}(M')$ is uniquely defined as an element of $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$. Thus, $\mathrm{Det}(M)$ is also uniquely determined. □

This uniqueness modulo commutators means that different valid sequences of elementary row operations used for triangularisation or diagonalisation will result in determinant values that belong to the same coset in $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$. For example, consider

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \tag{3.17}$$

over $\mathcal{D}$, assuming $a_{11}, a_{21} \neq 0$ for illustration of different pivot choices.

1. Using a_{11} as the chosen pivot, performing the row operation $R_2 \leftarrow R_2 - a_{21}a_{11}^{-1}R_1$ gives
$$\begin{pmatrix} a_{11} & a_{12} \\ 0 & a_{22} - a_{21}a_{11}^{-1}a_{12} \end{pmatrix}.$$
The determinant is $\mathrm{Det}_1 = [a_{11}(a_{22} - a_{21}a_{11}^{-1}a_{12})] = [a_{11}a_{22} - a_{11}a_{21}a_{11}^{-1}a_{12}]$.

2. Using a_{21} as pivot (first swap R_1, R_2, then $R_2 \leftarrow R_2 - a_{11}a_{21}^{-1}R_1$): The matrix becomes (after operations)
$$\begin{pmatrix} a_{21} & a_{22} \\ 0 & a_{12} - a_{11}a_{21}^{-1}a_{22} \end{pmatrix},$$
with a determinantal factor of $[-1]$ from the swap.
The determinant is $\mathrm{Det}_2 = [-1 \cdot a_{21}(a_{12} - a_{11}a_{21}^{-1}a_{22})] = [-a_{21}a_{12} + a_{21}a_{11}a_{21}^{-1}a_{22}]$.

It can be shown that Det_1 and Det_2 are equivalent modulo commutators. Specifically, expressions of the form $[X]$ and $[kX]$ (where k is a commutator) are equivalent, as are $[X]$ and $[Xk]$. The two forms above, Det_1 and Det_2, can be shown to differ by multiplication with elements whose class is [1] in $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$ (potentially including $[-1]$ if it is a commutator). The essential point is that they are equivalent in the quotient group.

Remark 3.6.6. *Throughout this study, all computations and statements regarding the Dieudonné determinant are understood to yield results modulo commutators, representing unique classes in* $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$ *(if invertible), or* 0 *(if singular).*

For matrices with entries in an Ore polynomial ring $\mathcal{S} = \mathcal{F}[\theta; \sigma]$ (that is $\mathcal{F}$ is a skew field, σ an automorphism, and $\delta = 0$), These matrices considered over the skew field of fractions $\mathcal{D} = \mathcal{F}(\theta; \sigma)$.

Lemma 3.6.7. *The Dieudonné determinant of a matrix* $M \in M_k(\mathcal{F}[\theta; \sigma])$ *(where* $\mathcal{F}$ *is a skew field and* σ *is an automorphism) can be represented by a skew polynomial in* $\mathcal{F}[\theta; \sigma]$*, modulo commutators from the skew field of fractions* $\mathcal{F}(\theta; \sigma)$*.*

Proof. Let $M \in M_k(\mathcal{F}[\theta; \sigma])$. If M is not invertible when considered over the skew field of fractions $\mathcal{F}(\theta; \sigma)$ then its Dieudonné determinant $\mathrm{Det}(M) = 0$ which is the zero polynomial. Assume M is invertible. Let M' be a copy of M which will be transformed into an upper triangular matrix using elementary row operations that preserve membership in $\mathcal{F}[\theta; \sigma]$ for the entries.

The transformation proceeds column by column. For each column i of M' (from 1 to $k - 1$):

1. **Pivot Selection:** If m'_{ii} is zero or an element of higher degree than some m'_{ji} $(j > i)$ in the current column i, use elementary row swaps to bring an element of minimal non-negative degree from rows $j \geq i$ in column i to the pivot position (i, i). Let this new pivot be m'_{ii}. If all m'_{ji} $(j \geq i)$ are zero, this column part is already processed. Row swaps affect $\mathrm{Det}(M)$ by at most a factor of $[-1]$, which is incorporated into the commutator class.

2. **Elimination Below Pivot:** Since $\mathcal{F}[\theta; \sigma]$ is a Euclidean domain (a known property, see e.g. [111] or [100]), for each entry m'_{ji} in column i where $j > i$: if $m'_{ji} \neq 0$, Euclidean (right) division of m'_{ji} can be performed by the pivot m'_{ii}. This yields $m'_{ji} = q \cdot m'_{ii} + r$, where $q, r \in \mathcal{F}[\theta; \sigma]$ and either $r = 0$ or $\deg(r) < \deg(m'_{ii})$. The elementary row operation $R_j \leftarrow R_j - qR_i$ then replaces the entry m'_{ji} with r. This process is repeated, reducing the degree

of m'_{ji} at each step, until m'_{ji} becomes 0. Operations of adding a multiple of one row to another do not change the Dieudonné determinant.

After applying this process for all necessary columns, M' is transformed into an upper triangular matrix (say T) whose entries t_{uv} (and particularly the diagonal entries t_{ii}, which are the final pivots) remain in $\mathcal{F}[\theta;\sigma]$.

As established (e.g. in Remark 3.6.3 or by the properties of Dieudonné determinants for triangular matrices), the determinant of T is the product of its diagonal entries (modulo commutators and a possible overall sign factor). Thus,

$$\mathrm{Det}(M) = \mathrm{Det}(T) = [\pm \prod_{i=1}^{k} t_{ii}]. \tag{3.18}$$

Since each diagonal entry t_{ii} is a polynomial in $\mathcal{F}[\theta;\sigma]$ and this ring is closed under multiplication, their product $\prod t_{ii}$ is also a skew polynomial in $\mathcal{F}[\theta;\sigma]$. Therefore, $\mathrm{Det}(M)$ is represented by this skew polynomial, considered as an element in $\mathcal{F}(\theta;\sigma)^{\times}/[\mathcal{F}(\theta;\sigma)^{\times},\mathcal{F}(\theta;\sigma)^{\times}]$ or as 0. □

Remark 3.6.8. *The Dieudonné determinant is formally defined for matrices over a skew field. The proof above operates by performing transformations within the ring $\mathcal{F}[\theta;\sigma]$, utilising its Euclidean domain structure. The resulting polynomial product $\prod t_{ii}$ is an element of $\mathcal{F}[\theta;\sigma]$. When this product is non-zero, it serves as a representative for the class of* $\mathrm{Det}(M)$ *in $\mathcal{F}(\theta;\sigma)^{\times}/[\mathcal{F}(\theta;\sigma)^{\times},\mathcal{F}(\theta;\sigma)^{\times}]$, where $\mathcal{F}(\theta;\sigma)$ is the Ore skew field of fractions of $\mathcal{F}[\theta;\sigma]$ (see, e.g. [33, Corollary 0.7.2]). The degree of elements in $\mathcal{F}(\theta;\sigma)$ can be defined, for instance, as* $\deg(g^{-1}f) = \deg(f) - \deg(g)$ *for $f, g \in \mathcal{F}[\theta;\sigma], g \neq 0$. The key result is that this determinantal class has a polynomial representative from $\mathcal{F}[\theta;\sigma]$.*

Based on the preceding discussion, the following lemma, theorem and remark can be stated:

Lemma 3.6.9 (Determinant as a Polynomial, cf. ([113], Lemma 11))**.** *The Dieudonné determinant of a matrix $A \in (\mathcal{F}[\theta;\sigma])^{k\times k}$ (where $\delta = 0$), if non-zero,*

can be represented by a unique monic skew polynomial in $\mathcal{F}[\theta;\sigma]$ *(up to multiplication by elements from* $[\mathcal{F}(\theta;\sigma)^\times, \mathcal{F}(\theta;\sigma)^\times]$ *that are also in* $\mathcal{F}[\theta;\sigma]$ *and are units). More practically, it is an element of* $\mathcal{F}[\theta;\sigma]$ *modulo such commutators.*

Theorem 3.6.10 (Determinant as a Polynomial via Triangularisation)**.** *Let* $\mathcal{S} = \mathcal{F}[\theta;\sigma]$ *be an Ore polynomial ring over a skew field* $\mathcal{F}$ *with* σ *an automorphism (so* $\mathcal{S}$ *is a Euclidean domain). Any invertible matrix* $A \in \mathcal{S}^{k\times k}$ *can be transformed into an upper (or a lower) triangular matrix* $T \in \mathcal{S}^{k\times k}$ *using elementary row operations within* $\mathcal{S}$*. The diagonal entries* $t_{11}, \ldots, t_{kk}$ *of* T *are non-zero elements of* $\mathcal{S}$*. The Dieudonné determinant is then*

$$\mathrm{Det}(A) = [\prod_{i=1}^{k} t_{ii}] \pmod{[\mathcal{F}(\theta;\sigma)^\times, \mathcal{F}(\theta;\sigma)^\times]}.$$

The product $\prod t_{ii}$ *is an element of* $\mathcal{S}$*.*

Remark 3.6.11 (Computation via Triangularisation)**.** *The computation of the Dieudonné determinant for* $A \in (\mathcal{F}[\theta;\sigma])^{k\times k}$ *typically involves triangularising the matrix using row operations within* $\mathcal{F}[\theta;\sigma]$ *such that diagonal entries remain polynomials. The Dieudonné determinant is then their product (modulo commutators and sign, if ignored). For practical purposes in resultant computation, often a specific representative polynomial from the commutator class is chosen.*

3.6.2 Hermite Form and Canonical Resultant Representatives

The Hermite form is a canonical matrix representation, applicable to matrices whose entries belong to a principal ideal domain (PID) or a Euclidean domain. This includes many commutative polynomial rings as well as non-commutative Ore polynomial rings such as $\mathcal{F}[\theta;\sigma]$ (where $\mathcal{F}$ is a skew field and σ an automorphism) [64]. For a matrix M over such a ring, its (row) Hermite form H is *unique* and shares the same row space as M (i.e. $H = UM$ for some unimodular matrix U). The Hermite form H satisfies the following properties (e.g. [114]):

(i) It is an upper triangular matrix.

(ii) The diagonal entries t_{ii} are monic (i.e. their leading coefficients are $1_{\mathcal{F}}$), or are zero. If M is invertible, all t_{ii} are non-zero and monic.

(iii) For each column j, if $t_{jj} \neq 0$, then all off-diagonal entries t_{ij} in that column (where $i < j$) have a degree strictly less than the degree of the diagonal entry t_{jj}. If $t_{jj} = 0$, then all t_{ij} for $i < j$ are also zero.

3.6.3 Filtered Rings

Although the Dieudonné determinant (and thus the skew resultant) is generally unique only up to commutators, its interpretation becomes more direct within the specific context of *almost commutative rings*. This concept is formalised by analysing the ring via a *filtration* and its *associated graded ring*, a technique common in the study of differential and difference operator algebras [104].

Definition 3.6.12 (Filtered Ring, see for example [114])**.** *A ring $\mathcal{S}$ (not necessarily commutative) is called* filtered *if there exists an ascending sequence of additive subgroups $\mathcal{S}_i$ ($i \in \mathbb{Z}_{\geq 0}$):*

$$\{0\} = \mathcal{S}_{-1} \subseteq \mathcal{S}_0 \subseteq \mathcal{S}_1 \subseteq \mathcal{S}_2 \subseteq \dots \quad \textit{such that} \quad \bigcup_{i \in \mathbb{Z}_{\geq 0}} \mathcal{S}_i = \mathcal{S},$$

and $1_{\mathcal{S}} \in \mathcal{S}_0$, and additionally $\mathcal{S}_i \, \mathcal{S}_j \subseteq \mathcal{S}_{i+j}$ for all $i, j \in \mathbb{Z}_{\geq 0}$. This makes $\mathcal{S}_0$ a subring of $\mathcal{S}$. A common filtration for a polynomial ring $\mathcal{S} = \mathcal{F}[\theta_1, \dots, \theta_n]$ is by total degree, where $\mathcal{S}_i$ consists of polynomials of total degree at most i.

Definition 3.6.13 (Associated Graded Ring)**.** *Let $\mathcal{S} = \bigcup_{i \in \mathbb{Z}_{\geq 0}} \mathcal{S}_i$ be a filtered ring. The* associated graded ring *of $\mathcal{S}$, denoted $\operatorname{gr}(\mathcal{S})$, is defined as the direct sum of factor groups:*

$$\operatorname{gr}(\mathcal{S}) = \bigoplus_{i \in \mathbb{Z}_{\geq 0}} \operatorname{gr}_i(\mathcal{S}) \quad \textit{where} \quad \operatorname{gr}_i(\mathcal{S}) = \mathcal{S}_i \,/\, \mathcal{S}_{i-1} \ \textit{(with } \mathcal{S}_{-1} = \{0\}\textit{)}.$$

Multiplication in $\mathrm{gr}(\mathcal{S})$ *is defined for homogeneous elements* $\bar{r} = r + \mathcal{S}_{i-1} \in \mathrm{gr}_i(\mathcal{S})$ *and* $\bar{s} = s + \mathcal{S}_{j-1} \in \mathrm{gr}_j(\mathcal{S})$ *by*

$$\bar{r}\bar{s} = (rs + \mathcal{S}_{i+j-1}) \in \mathrm{gr}_{i+j}(\mathcal{S}).$$

For any $r \in \mathcal{S}$*, if* $r \in \mathcal{S}_i \setminus \mathcal{S}_{i-1}$ *(i.e.* r *has "filtered degree" i), its principal symbol* $\widetilde{\sigma}(r)$ *is its image* $r + \mathcal{S}_{i-1}$ *in* $\mathrm{gr}_i(\mathcal{S})$*. If* $\widetilde{\sigma}(r)\widetilde{\sigma}(s) \neq 0$ *in* $\mathrm{gr}(\mathcal{S})$*, then*

$$\widetilde{\sigma}(rs) = \widetilde{\sigma}(r)\widetilde{\sigma}(s). \tag{3.19}$$

Definition 3.6.14 (Almost Commutative Ring, see for example [114])**.** *A filtered ring* $\mathcal{S}$ *is called* almost commutative *if its associated graded ring* $\mathrm{gr}(\mathcal{S})$ *is commutative [83, §3.3]. Many rings of operators, such as Weyl algebras or universal enveloping algebras, are almost commutative.*

Chapter 4

Resultant-Based Elimination for Skew Polynomial Systems

Many models in mathematical physics, engineering, and computer science involve systems of equations with multiple parameters or indeterminates. A fundamental task is often to simplify such systems by eliminating one or more indeterminates, while preserving essential properties or relationships among the remaining ones. This process of elimination can transform a complex system into a more manageable form, facilitating analysis, solution, or further computation. Resultant-based methods provide a powerful algebraic toolkit for achieving such elimination.

Historically, resultants, such as the Sylvester resultant, were developed for commutative polynomials to determine conditions under which two polynomials share a common root, without explicitly computing the roots. This concept naturally extends to an elimination tool in the commutative settings; the resultant of $f(x, y)$ and $g(x, y)$ taken with respect to x, denoted $\mathrm{Res}_x(f, g)$, is a polynomial solely in y that vanishes for any specialisation of y causing f and g (viewed as polynomials in x) to share a common root.

Extending these ideas to non-commutative polynomial rings, particularly Ore polynomial rings, presents significant challenges due to the non-commutative multiplication, the behaviour of evaluation, and the definition of determinants. Oys-

tein Ore himself investigated determinants for systems of linear equations over skew fields in his foundational work [110]. Ore's approach focused on solving linear systems $AX = B$. For a 2×2 system:

$$a_{11}X_1 + a_{12}X_2 = b_1$$
$$a_{21}X_1 + a_{22}X_2 = b_2$$

where a_{ij}, b_i are in a skew field (or an Ore ring embeddable in one), Ore's method involves finding f_{12}, f_{22} such that $f_{22}a_{12} = f_{12}a_{22}$ (a common multiple relation). Then, one forms $(f_{22}a_{11} - f_{12}a_{21})X_1 = f_{22}b_1 - f_{12}b_2$. The term $(f_{22}a_{11} - f_{12}a_{21})$ is considered an Ore determinant. However, as observed by Ore, the elements f_{12}, f_{22} can be chosen non-uniquely, yielding *different determinant* expressions with potentially different degrees. Ore noted that for his purposes, it was sufficient to know if the determinant vanished or not, implying equivalence if they co-vanish [110]. This non-uniqueness, especially of degree, makes Ore's original determinant concept less suitable for defining a resultant polynomial whose degree and structure are critical for the applications discussed in this study.

To establish a more effective and well-defined resultant for skew polynomials, this work, using new approaches in non-commutative algebra by employing the Dieudonné determinant (as detailed in Chapter 3, Section 3.6), the Dieudonné determinant provides a consistent mapping from matrices over a skew field to an abelian group (the multiplicative group of the skew field modulo its commutator subgroup), or to zero if the matrix is singular. This chapter details resultant-based approaches for eliminating indeterminates in bivariate skew polynomial systems, generalising classical methods and utilising the Dieudonné determinant for this purpose.

4.1 The Resultant of Bivariate Skew Polynomials

Let $f(\theta_1, \theta_2)$ and $g(\theta_1, \theta_2)$ be two bivariate skew polynomials belonging to the iterated Ore polynomial ring $\mathcal{S} = \mathcal{F}[\theta_1; \sigma_1, \delta_1][\theta_2; \sigma_2, \delta_2]$ over a (skew) field $\mathcal{F}$.

Let $\mathcal{R}_1 = \mathcal{F}[\theta_1; \sigma_1, \delta_1]$ denote the ring of coefficients. For simplicity in the following arguments, the derivations are assumed to be zero ($\delta_1 = 0, \delta_2 = 0$). The ring is thus $\mathcal{S} = \mathcal{F}[\theta_1; \sigma_1][\theta_2; \sigma_2]$, with commutation rules $\theta_1 a = \sigma_1(a)\theta_1$ for $a \in \mathcal{F}$, and $\theta_2 c = \sigma_2(c)\theta_2$ for $c \in \mathcal{R}_1$.

Recall, both f and g can be written as polynomials in θ_2 with coefficients in $\mathcal{R}_1$:

$$f(\theta_1, \theta_2) = \sum_{i=0}^{m} a_i(\theta_1)\theta_2^i, \quad \text{with } a_m(\theta_1) \neq 0, \quad m = \deg_{\theta_2}(f)$$
$$g(\theta_1, \theta_2) = \sum_{j=0}^{k} b_j(\theta_1)\theta_2^j, \quad \text{with } b_k(\theta_1) \neq 0, \quad k = \deg_{\theta_2}(g).$$

The resultant $\mathrm{Res}_{\theta_2}(f, g)$ eliminates the indeterminate θ_2 by providing a condition in the ring $\mathcal{R}_1$ for the polynomials f and g to possess a common right divisor in $\mathcal{S}$.

4.1.1 Generalisation of the Sylvester Matrix

Intuitively, the classical Sylvester matrix encodes a system of linear equations designed to detect whether two polynomials share a common factor. By arranging shifted copies of the polynomials (their multiples by powers of the indeterminate) into rows, any non-trivial linear dependence among these rows implies a relationship of the form $Af + Bg = 0$. In the skew setting, this fundamental logic is preserved, but the construction must account for non-commutativity. Since left-multiplying a polynomial by θ_2^k has the effect of applying the automorphism σ_2^k to its coefficients, the rows of the matrix must reflect this *twisted* structure. Consequently, the resulting σ-Sylvester matrix captures the conditions for a common right divisor within the non-commutative ring, ensuring that the singularity

of the matrix corresponds to an algebraic dependence between the polynomials.

The algebraic condition for f and g to share a common factor can be derived in two equivalent methods. Both derivations lead to the construction of the same system of linear equations and its corresponding coefficient matrix. This matrix, denoted as the *σ-Sylvester matrix* $\mathrm{Sylv}_{\theta_2}(f, g)$, extends the classical Sylvester matrix to the Ore algebra setting $\mathcal{F}[\theta_1; \sigma_1][\theta_2; \sigma_2]$ by incorporating the non-commutative multiplication rules, as formally defined below.

Definition 4.1.1 (σ-Sylvester Matrix for Skew Polynomials)**.** *Let $f = \sum_{i=0}^{m} a_i \theta_2^i$ and $g = \sum_{j=0}^{k} b_j \theta_2^j$ be polynomials in $\mathcal{S} = (\mathcal{F}[\theta_1; \sigma_1])[\theta_2; \sigma_2]$. The σ-Sylvester matrix, $\mathrm{Sylv}_{\theta_2}(f, g)$, is an $(m+k) \times (m+k)$ matrix whose entries are from $\mathcal{F}[\theta_1; \sigma_1]$. Its rows are constructed from the coefficients of the polynomials $\{\theta_2^{k-1} f, \ldots, f\}$ and $\{\theta_2^{m-1} g, \ldots, g\}$. Using the notation $c^{[i]} = \sigma_2^i(c)$, the matrix has the following block structure:*

$$\mathrm{Sylv}_{\theta_2}(f, g) = \begin{pmatrix} a_m^{[k-1]} & a_{m-1}^{[k-1]} & \cdots & a_0^{[k-1]} & & \\ & a_m^{[k-2]} & \cdots & a_1^{[k-2]} & a_0^{[k-2]} & \\ & & \ddots & & & \ddots \\ & & & a_m^{[0]} & \cdots & a_0^{[0]} \\ b_k^{[m-1]} & b_{k-1}^{[m-1]} & \cdots & b_0^{[m-1]} & & \\ & b_k^{[m-2]} & \cdots & b_1^{[m-2]} & b_0^{[m-2]} & \\ & & \ddots & & & \ddots \\ & & & b_k^{[0]} & \cdots & b_0^{[0]} \end{pmatrix}$$

The following subsections briefly outline how different algebraic starting points both lead to this same construction.

4.1.1.1 Motivation from the Common Factor Condition

The classical approach states that f and g have a common factor if and only if there exist non-zero polynomials $P, Q \in \mathcal{S}$ satisfying the Bézout-like identity $Pf + Qg = 0$, with bounded degrees; $\deg_{\theta_2}(P) < k$ and $\deg_{\theta_2}(Q) < m$. This identity can be expanded and rewritten as a system of $m+k$ linear homogeneous equations by collecting the coefficients of corresponding powers of θ_2. The condition for the existence of a non-trivial solution for the unknown coefficients of P and Q is that the determinant of this linear system must be zero.

4.1.1.2 Motivation From the Least Common Left Multiple

An alternative derivation arises from finding the least common left multiple (lclm) of f and g. The existence of an lclm is equivalent to finding non-zero polynomials $U, V \in \mathcal{S}$ of minimal degree such that:

$$Uf = Vg. \tag{4.1}$$

The existence of the polynomials U and V, as well as the greatest common right divisor (gcrd), is a direct consequence of the Skew Extended Euclidean Algorithm (Algorithm 2), which is applicable to any two elements in the right-Euclidean domain $\mathcal{R}_1 = \mathcal{F}[\theta_1; \sigma_1]$.

The bivariate ring $\mathcal{S} = \mathcal{R}_1[\theta_2; \sigma_2, \delta_2]$, however, is not generally a Euclidean domain. Instead, the goal is to construct the coefficients of U and V in (4.1) directly. Thus, the aim is to identify a non-trivial solution with minimal degrees, namely $\deg_{\theta_2}(U) \leq k-1$ and $\deg_{\theta_2}(V) \leq m-1$. Consider

$$U(\theta_1, \theta_2) = \sum_{i=0}^{k-1} u_i(\theta_1)\theta_2^i$$
$$V(\theta_1, \theta_2) = \sum_{j=0}^{m-1} v_j(\theta_1)\theta_2^j.$$

The equation $Uf = Vg$ becomes:

$$\left(\sum_{i=0}^{k-1} u_i \theta_2^i\right)\left(\sum_{j=0}^{m} a_j \theta_2^j\right) = \left(\sum_{i=0}^{m-1} v_i \theta_2^i\right)\left(\sum_{j=0}^{k} b_j \theta_2^j\right). \tag{4.2}$$

To equate coefficients, the products must first be expressed in the standard basis $\{1, \theta_2, \ldots\}$. This involves applying the commutation rule $\theta_2 c = \sigma_2(c)\theta_2$. Generally, $\theta_2^i c = \sigma_2^i(c)\theta_2^i$. The equation becomes:

$$\sum_{i=0}^{k-1}\sum_{j=0}^{m} u_i a_j^{[i]} \theta_2^{i+j} = \sum_{i=0}^{m-1}\sum_{j=0}^{k} v_i b_j^{[i]} \theta_2^{i+j}.$$

Comparing the coefficients of θ_2^p for $p = 0, \ldots, m+k-1$ on both sides forms a system of $m+k$ homogeneous linear equations for the $m+k$ unknown coefficients $u_0, \ldots, u_{k-1}$ and $v_0, \ldots, v_{m-1}$. This can be written as:

$$\begin{cases} u_0 a_0^{[0]} - v_0 b_0^{[0]} = 0 & (\text{coeff of } \theta_2^0) \\ u_0 a_1^{[0]} + u_1 a_0^{[1]} - v_0 b_1^{[0]} - v_1 b_0^{[1]} = 0 & (\text{coeff of } \theta_2^1) \\ \qquad \vdots & \\ u_{k-1} a_m^{[k-1]} - v_{m-1} b_k^{[m-1]} = 0 & (\text{coeff of } \theta_2^{m+k-1}) \end{cases} \tag{4.3}$$

For a non-trivial solution for the u_i and v_j to exist, the system's coefficient matrix must be singular. The matrix of this system is the σ-Sylvester matrix, whose determinant can be formulated as follows.

$$\mathrm{Det}(\mathrm{Sylv}_{\theta_2}(f,g)) = \begin{vmatrix} a_m^{[k-1]} & a_{m-1}^{[k-1]} & \cdots & \cdots & a_0^{[k-1]} & & \\ & a_m^{[k-2]} & a_{m-1}^{[k-2]} & \cdots & \cdots & a_0^{[k-2]} & \\ & & \ddots & & & & \ddots \\ & & & a_m^{[0]} & a_{m-1}^{[0]} & \cdots & \cdots & a_0^{[0]} \\ b_k^{[m-1]} & b_{k-1}^{[m-1]} & \cdots & \cdots & b_0^{[m-1]} & & \\ & b_k^{[m-2]} & b_{k-1}^{[m-2]} & \cdots & \cdots & b_0^{[m-2]} & \\ & & \ddots & & & \ddots & \\ & & & b_k^{[0]} & b_{k-1}^{[0]} & \cdots & \cdots & b_0^{[0]} \end{vmatrix}$$

The entries of this matrix are univariate skew polynomials in θ_1 from the non-commutative ring $\mathcal{R}_1 = \mathcal{F}[\theta_1; \sigma_1]$. To evaluate this determinant, the Dieudonné determinant is used, which requires embedding $\mathcal{R}_1$ in its skew field of fractions $\mathcal{K} = \mathcal{F}(\theta_1; \sigma_1)$.

4.1.2 The Skew Resultant

At an intuitive level, the skew resultant is defined to act as a precise algebraic characterisation of elimination, mirroring its role in the commutative case; it is an object whose vanishing provides a necessary and sufficient condition for two skew polynomials f and g to share a non-trivial common right factor. However, unlike the commutative resultant, which yields a unique scalar, the skew resultant must accommodate the lack of commutativity in the coefficient domain. This is achieved by employing the Dieudonné determinant of the σ-Sylvester matrix. While this determinant is defined only up to commutator factors, its property of being zero or non-zero is invariant, thereby successfully encoding the common-factor compatibility condition into the underlying coefficient skew field (modulo commutators).

In the previous formulation, both derivations converge on the same condition: the singularity of the σ-Sylvester matrix. This is formalised by employing the Dieudonné determinant.

Definition 4.1.2 (Resultant of Bivariate Skew Polynomials)**.** *Let* $f = \sum_{i=0}^{m} a_i(\theta_1)\theta_2^i$ *and* $g = \sum_{j=0}^{k} b_j(\theta_1)\theta_2^j$ *be elements of the bivariate skew polynomial ring* $\mathcal{S} = (\mathcal{F}[\theta_1; \sigma_1])[\theta_2; \sigma_2]$*, with coefficients* $a_i(\theta_1), b_j(\theta_1)$ *from* $\mathcal{F}[\theta_1; \sigma_1]$*. The resultant of* f *and* g *with respect to* θ_2*, denoted* $\mathrm{Res}_{\theta_2}(f, g)$*, is defined over*

the skew field of fractions $\mathcal{K} = \mathcal{F}(\theta_1; \sigma_1)$ *as follows:*

$$\mathrm{Res}_{\theta_2}(f,g) = \begin{vmatrix}
\theta_2^{k-1}f & a_m^{[k-1]} & a_{m-1}^{[k-1]} & \cdots & \cdots & a_0^{[k-1]} & & & \\
\theta_2^{k-2}f & & a_m^{[k-2]} & a_{m-1}^{[k-2]} & \cdots & \cdots & a_0^{[k-2]} & & \\
\vdots & & & \ddots & & & & \ddots & \\
f & & & & a_m^{[0]} & a_{m-1}^{[0]} & \cdots & \cdots & a_0^{[0]} \\
\theta_2^{m-1}g & b_k^{[m-1]} & b_{k-1}^{[m-1]} & \cdots & \cdots & b_0^{[m-1]} & & & \\
\theta_2^{m-2}g & & b_k^{[m-2]} & b_{k-1}^{[m-2]} & \cdots & \cdots & b_0^{[m-2]} & & \\
\vdots & & & \ddots & & & \ddots & & \\
g & & & & b_k^{[0]} & b_{k-1}^{[0]} & \cdots & \cdots & b_0^{[0]}
\end{vmatrix},$$

where the i*-th row (for* $i = 1, \ldots, k$*) contains the coefficient sequence of the multiplication* $\theta_2^{k-i}f$*. Similarly, the* $(k+j)$*-th row (for* $j = 1, \ldots, m$*), contains the coefficients of* $\theta_2^{m-j}g$*. For notational simplicity, it is convenient to write:*

$$\mathrm{Res}_{\theta_2}(f,g) = \mathrm{Det}(\theta_2^{k-1}f, \ldots, \theta_2 f, f, \theta_2^{m-1}g, \ldots, \theta_2 g, g). \tag{4.4}$$

This definition extends concepts from [52] to the bivariate skew case using the Dieudonné determinant which can be represented by a polynomial utilising the fact that the Dieudonné determinant of a matrix with entries in $\mathcal{F}[\theta_1; \sigma_1]$ can be represented as a polynomial in $\mathcal{F}[\theta_1; \sigma_1]$ as established in Lemma 3.6.9 and Theorem 3.6.10.

The computation of this Dieudonné determinant typically involves techniques such as Gaussian elimination adapted for skew polynomial entries, as discussed in Section 3.6.

Recall, the non-commutative multiplication is determined by the rules

$$\theta_1 a = \sigma_1(a)\theta_1 \text{ and } \theta_2 c = \sigma_2(c)\theta_2,$$

which ensures the algebraic structure is respected correctly during the σ-Sylvester matrix construction where rows are left-multiplied by powers of θ_2.

The primary distinction between this definition (Definition 4.1.2) and that used in [52] is the focus on *bivariate* skew polynomials with *two commuting indeterminates*, a formulation particularly suited to the Ore algebra framework used in this study. This represents a new combination that facilitates the elimination of indeterminates in the general multivariate case ($n > 1$) and enables the use of an *operator evaluation map*; a central theme of this research.

Remark 4.1.3. *The vanishing of the resultant,* $\mathrm{Res}_{\theta_2}(f, g) = 0$*, is both a necessary and sufficient condition for the* σ*-Sylvester matrix to be singular over* $\mathcal{K}$*. This singularity, in turn, indicates the existence of non-trivial polynomials* U *and* V *such that* $Uf = Vg$*. Consequently,* $\mathrm{Res}_{\theta_2}(f, g) = 0$ *if and only if* f *and* g *share a common right factor in* $\mathcal{S}$*.*

4.2 The Direct Resultant Algorithm

The direct method for computing the skew resultant, based on the principles discussed above, which provides a constructive approach to elimination without relying on indeterminate evaluation. To ensure clarity regarding the operational logic of the subsequent algorithm, a conceptual overview is provided below.

Conceptual Overview

The algorithm proceeds in three distinct phases: matrix construction, unimodular reduction, and determinant extraction.

1. **Construction of the σ-Sylvester Matrix:** The process begins by linearising the problem. The σ-Sylvester matrix S associated with the input polynomials f and g is constructed. As defined previously, the rows of S represent the coefficients of the polynomials $\theta_2^k f$ and $\theta_2^j g$. Crucially, the entries of this matrix lie within the univariate skew polynomial ring $\mathcal{R}_1 = \mathcal{F}[\theta_1; \sigma_1]$.

This step translates the condition of a common right divisor in the bivariate ring into a linear dependency problem over the coefficient ring $\mathcal{R}_1$.

2. **Unimodular Triangularisation:** The core of the algorithm is the transformation of S into an upper triangular form T. Unlike standard Gaussian elimination over a field, which introduces fractions, this algorithm operates over the ring $\mathcal{R}_1$ (a Euclidean domain). To avoid the coefficient swell associated with fraction arithmetic, *unimodular row operations* are employed. For any two entries p and q in a column, a specific linear combination is computed that annihilates one of them. This is achieved by finding left annihilators (u, v) such that $up+vq = 0$, derived from the extended Euclidean algorithm in $\mathcal{R}_1$. These operations correspond to left-multiplication by unimodular matrices (matrices with invertible determinants in $\mathcal{F}$), thereby preserving the Dieudonné determinant of the system up to a unit factor.

3. **Resultant Extraction:** Once the matrix T is in upper triangular form, the Dieudonné determinant is given by the product of the diagonal elements. Due to the non-commutative nature of $\mathcal{R}_1$, the order of multiplication is preserved as it appears on the diagonal (from top-left to bottom-right). The resulting product $R(\theta_1)$ is the computed skew resultant.

The detailed procedural steps implementing this logic are outlined in Algorithm 3.

Algorithm 3: Direct Resultant Computation via Unimodular Triangularisation

Input : Two bivariate skew polynomials $f(\theta_1, \theta_2)$ and $g(\theta_1, \theta_2)$ belonging to the ring $\mathcal{S} = \mathcal{F}[\theta_1; \sigma_1][\theta_2; \sigma_2]$.

Output: The resultant polynomial $R(\theta_1) = \mathrm{Res}_{\theta_2}(f, g)$ belonging to the coefficient ring $\mathcal{R}_1 = \mathcal{F}[\theta_1; \sigma_1]$.

/* Step 1: Construct the σ-Sylvester Matrix */

$m := \deg_{\theta_2}(f)$; $n := \deg_{\theta_2}(g)$; $N := m + n$

Build Sylvster matrix $S := \mathrm{Sylv}_{\theta_2}(f, g) \in M_N(\mathcal{R}_1)$

/* Rows $1, \ldots, n$ are formed from coeffs. of $\theta_2^{n-1} f, \ldots, \theta_2^0 f$ */

/* Rows $n+1, \ldots, N$ are from coeffs. of $\theta_2^{m-1} g, \ldots, \theta_2^0 g$ */

$T := S$

// Initialise the matrix T for triangularisation

/* Step 2: Triangularise T via unimodular row operations */

for $j := 1$ **to** $N - 1$ // pivot column

do

  if $T_{j,j} = 0$ **then** swap row j with first $k > j$ having $T_{k,j} \neq 0$

  for $k := j + 1$ **to** N // eliminate below pivot

  do

    if $T_{k,j} \neq 0$ **then**

      $p := T_{j,j}$; $q := T_{k,j}$

      $(u, v) :=$ annihilators(p, q)

      // $up + vq = 0$

      for $c := j$ **to** N **do**

        $T_{k,c} := u \cdot T_{j,c} + v \cdot T_{k,c}$

      end

    end

  end

end

The matrix T is now upper triangular

/* Step 3: Compute Resultant from Diagonal Product */

Let $d_k := T_{k,k}$ (for $k = 1, \ldots, N$) be the diagonal entries of T

The resultant is the ordered non-commutative product:

$$R(\theta_1) = d_1 \star d_2 \star \cdots \star d_N$$

where $\star$ denotes multiplication in the Ore ring $\mathcal{R}_1$

/* Step 4: Return result */

return $R(\theta_1)$

4.2.1 Resultant Degree and Commutator Factors

While the Dieudonné determinant, and thus the skew polynomial resultant derived from it, is unique only up to multiplication by commutators, its degree as a polynomial is well-defined. This was noted by Taelman [122], who observed that the degree map is zero on commutators, ensuring the invariance of the resultant's degree.

The involvement of commutator factors is an inherent feature of elimination in non-commutative settings. This can be illustrated with a simple illustrative system of two linear equations in an indeterminate θ over a (skew) field:

$$\begin{cases} a_1\theta + a_0 &= 0 \\ b_1\theta + b_0 &= 0 \end{cases} \tag{4.5}$$

where $a_0, a_1, b_0, b_1 \in \mathcal{F}$ and $a_1, b_1 \neq 0$. For eliminating θ, we can multiply these equations by a_1 and $-b_1$ respectively to obtain polynomial expressions:

1. Elimination via the first equation yields the condition $b_0 - b_1 a_1^{-1} a_0 = 0$. Left-multiplying by a_1 gives the expression:

$$R_A = a_1(b_0 - b_1 a_1^{-1} a_0) = a_1 b_0 - a_1 b_1 a_1^{-1} a_0. \tag{4.6}$$

2. Elimination via the second equation yields $a_0 - a_1 b_1^{-1} b_0 = 0$. Left-multiplying by $-b_1$ gives the expression:

$$R_B = -b_1(a_0 - a_1 b_1^{-1} b_0) = b_1 a_1 b_1^{-1} b_0 - b_1 a_0. \tag{4.7}$$

The two expressions, R_A and R_B, are not generally identical. However, they are equivalent modulo commutators, as one can be shown to be a commutator multiple of the other. Let $\mathbf{c} = a_1 b_1 a_1^{-1} b_1^{-1}$ be the commutator of a_1 and b_1. The equivalence can be demonstrated by manipulating the expression for R_A as

follows:

$$\begin{aligned}
R_A &= a_1 b_0 - a_1 b_1 a_1^{-1} a_0 \\
&= a_1 b_0 - (a_1 b_1 a_1^{-1} b_1^{-1}) b_1 a_0 \quad \text{(inserting } b_1^{-1} b_1 = 1\text{)} \\
&= a_1 b_0 - \mathbf{c} \cdot b_1 a_0 \\
&= \mathbf{c}(\mathbf{c}^{-1} a_1 b_0 - b_1 a_0) \\
&= \mathbf{c}((b_1 a_1 b_1^{-1} a_1^{-1}) a_1 b_0 - b_1 a_0) \\
&= \mathbf{c}(b_1 a_1 b_1^{-1} b_0 - b_1 a_0) \\
&= \mathbf{c} \cdot R_B.
\end{aligned}$$

Since $R_A = \mathbf{c} \cdot R_B$, their classes in the abelian group $\mathcal{F}^\times/[\mathcal{F}^\times, \mathcal{F}^\times]$ are equivalent, as $[\mathbf{c}] = [1]$. This example illustrates that the resultant condition is unique only up to these commutator factors. The consistent definition of the resultant via the Dieudonné determinant correctly handles these non-commutative aspects by providing a single, canonical value in this quotient group.

From this viewpoint, a central difference between commutative and non-commutative elimination is that a determinant, and hence a determinant-based resultant, is no longer a well-defined scalar in the coefficient skew field, but only well-defined *up to multiplication by commutators*. This commutator ambiguity is precisely what is meant by a *commutator factor*.

Let D be a skew field and let $D^\times$ denote its group of units. The *commutator subgroup* is

$$[D^\times, D^\times] \;=\; \langle\, aba^{-1}b^{-1} \;:\; a, b \in D^\times \,\rangle.$$

For matrices over D, the appropriate determinant is the *Dieudonné determinant*

$$\det_D : \mathrm{GL}_n(D) \longrightarrow D^\times/[D^\times, D^\times],$$

whose value lies in the abelianised unit group.

This algebraic framework is crucial for understanding the properties of the

resultant, particularly regarding its degree.

When stating degree formulae for skew resultants (and related scaling/evaluation properties), the commutator factor explains *why* the resultant is only canonical in $D^\times/[D^\times, D^\times]$; nevertheless, any degree/valuation extracted from it is canonical because commutators have trivial degree. In particular, commutator factors do not inflate the resultant degree, but they must be accounted for to justify that the degree is independent of the chosen determinant representative or elimination sequence.

Example 4.2.1 (σ-Sylvester Matrix Construction)**.** *Let* $f(\theta_1, \theta_2) = A_2(\theta_1)\theta_2^2 + A_1(\theta_1)\theta_2 + A_0(\theta_1)$ *and* $g(\theta_1, \theta_2) = B_2(\theta_1)\theta_2^2 + B_1(\theta_1)\theta_2 + B_0(\theta_1)$*, where* $A_i(\theta_1), B_j(\theta_1) \in \mathcal{F}[\theta_1; \sigma_1]$*. Assume* $\delta_2 = 0$*. Here* $m = 2, k = 2$*. The* σ*-Sylvester matrix* $\mathrm{Sylv}_{\theta_2}(f, g)$ *is a* 4×4 *matrix. The polynomials involved are* $\theta_2 f, f, \theta_2 g, g$*.*

$$
\begin{aligned}
f &= A_2\theta_2^2 + A_1\theta_2 + A_0 \\
\theta_2 f &= \theta_2(A_2\theta_2^2 + A_1\theta_2 + A_0) = \sigma_2(A_2)\theta_2^3 + \sigma_2(A_1)\theta_2^2 + \sigma_2(A_0)\theta_2 \\
g &= B_2\theta_2^2 + B_1\theta_2 + B_0 \\
\theta_2 g &= \theta_2(B_2\theta_2^2 + B_1\theta_2 + B_0) = \sigma_2(B_2)\theta_2^3 + \sigma_2(B_1)\theta_2^2 + \sigma_2(B_0)\theta_2
\end{aligned}
$$

The σ*-Sylvester matrix, with columns corresponding to coefficients of* $\theta_2^3, \theta_2^2, \theta_2^1, \theta_2^0$*, is:*

$$
\mathrm{Sylv}_{\theta_2}(f, g) = \begin{pmatrix}
\sigma_2(A_2) & \sigma_2(A_1) & \sigma_2(A_0) & 0 \\
0 & A_2 & A_1 & A_0 \\
\sigma_2(B_2) & \sigma_2(B_1) & \sigma_2(B_0) & 0 \\
0 & B_2 & B_1 & B_0
\end{pmatrix}.
$$

The entries $A_i(\theta_1), B_j(\theta_1), \sigma_2(A_i(\theta_1)), \sigma_2(B_j(\theta_1))$ *are elements of* $\mathcal{F}[\theta_1; \sigma_1]$*.*

Example 4.2.2 (Direct Resultant Computation)**.** *Let* $\mathcal{F} = \mathbb{C}(t_1)$ *and consider* $\mathcal{S} = \mathcal{F}[\theta_1; \sigma_1][\theta_2; \sigma_2]$*, where* σ_1 *is complex conjugation on coefficients and* σ_2 *is*

the identity on $\mathcal{F}[\theta_1;\sigma_1]$. *Consider the polynomials:*

$$f(\theta_1,\theta_2) = (t_1 i\theta_1)\theta_2^2 + (t_2+1)$$
$$g(\theta_1,\theta_2) = (t_2 i\theta_1)\theta_2 + t_1 i$$

Here, $m = 2, k = 1$. *The* σ*-Sylvester matrix* $\mathrm{Sylv}_{\theta_2}(f,g)$ *is a* 3×3 *matrix formed from the coefficients of* f, $\theta_2 g$, *and* g. *With* $A_2 = t_1 i\theta_1, A_0 = t_2+1$ *and* $B_1 = t_2 i\theta_1, B_0 = t_1 i$, *and noting* σ_2 *is the identity* ($c^{[i]} = c$):

$$\mathrm{Sylv}_{\theta_2}(f,g) = \begin{pmatrix} A_2 & 0 & A_0 \\ B_1^{[1]} & B_0^{[1]} & 0 \\ 0 & B_1 & B_0 \end{pmatrix} = \begin{pmatrix} t_1 i\theta_1 & 0 & t_2+1 \\ t_2 i\theta_1 & t_1 i & 0 \\ 0 & t_2 i\theta_1 & t_1 i \end{pmatrix}.$$

The Dieudonné determinant $\mathrm{Det}(\mathrm{Sylv})$ *is computed. Following a Gaussian elimination with* $\sigma_2(t_2 i\theta_1) = (t_2+1)i\theta_1$ *(when* σ_2 *is shift on* t_2 *and* $\sigma_2(\theta_1) = \theta_1$, $\sigma_2(i) = i$). *That is* $\sigma_2(t_2 i) = (t_2+1)i$ *and* $\sigma_2(t_1 i) = t_1 i$:

$$M = \begin{pmatrix} t_1 i\theta_1 & 0 & t_2+1 \\ (t_2+1)i\theta_1 & t_1 i & 0 \\ 0 & t_2 i\theta_1 & t_1 i \end{pmatrix}.$$

Performing row operations to triangularise (details omitted here but involve non-commutative algebra carefully): $R_1 \leftrightarrow R_2$: *(sign change, or incorporate into commutator class)*

$$M_1 = \begin{pmatrix} (t_2+1)i\theta_1 & t_1 i & 0 \\ t_1 i\theta_1 & 0 & t_2+1 \\ 0 & t_2 i\theta_1 & t_1 i \end{pmatrix}.$$

$R_2 \leftarrow R_2 - (t_1 i\theta_1)((t_2+1)i\theta_1)^{-1} R_1 = R_2 - t_1(t_2+1)^{-1} R_1$. *For this to be polynomial, coefficients must be carefully handled. Assume calculations lead to*

upper triangular T:

$$T = \begin{pmatrix} d_1(\theta_1) & * & * \\ 0 & d_2(\theta_1) & * \\ 0 & 0 & d_3(\theta_1) \end{pmatrix}.$$

Then $\mathrm{Res}_{\theta_2}(f, g) = [d_1(\theta_1)d_2(\theta_1)d_3(\theta_1)]$. *The result, after correcting for the specific* σ_1, σ_2 *actions, is* $[(-t_2^3 - t_2^2 - t_1^2)\theta_1^2 - t_1^3\theta_1]$. *This example highlights that computation involves careful application of Ore polynomial arithmetic over* $\mathcal{F}(\theta_1; \sigma_1)$.

4.3 Properties of the Skew Polynomial Resultant

The skew resultant exhibits several essential properties analogous to the commutative resultant, which are essential for its application in elimination theory.

4.3.1 Uniqueness of the Resultant

Let M be a square matrix, the Dieudonné determinant $\mathrm{Det}(M)$ over a skew field $\mathcal{D}$ takes its value in the abelian group $\mathcal{D}^\times/[\mathcal{D}^\times, \mathcal{D}^\times]$ when M is invertible, and is 0 if M is singular. Consequently, the resultant $\mathrm{Res}_{\theta_2}(f, g)$ is uniquely defined as a class in the corresponding quotient group, $\mathcal{F}(\theta_1; \sigma_1)^\times/[\mathcal{F}(\theta_1; \sigma_1)^\times, \mathcal{F}(\theta_1; \sigma_1)^\times]$, or as 0.

As established in Lemma 3.6.9 and Theorem 3.6.10, this class has a representative that is a polynomial belong to $\mathcal{F}[\theta_1; \sigma_1]$. While this polynomial representative differs from another only by a factor that is a unit commutator, its degree is well-defined [122]. For many applications, any polynomial representative from this equivalence class is sufficient, or a canonical representative (e.g., monic) can be chosen.

4.3.1.1 Almost Commutative Rings

If $\mathcal{S}$ is an almost commutative ring (e.g. $\mathcal{F}[\theta_1; \sigma_1] \dots [\theta_n; \sigma_n]$ where σ_i are such that $\mathrm{gr}(\mathcal{S})$ is commutative, often $\mathcal{F}[\theta_1, \dots, \theta_n]$ with standard degree filtration)

and $\mathrm{gr}(\mathcal{S})$ is a UFD, then the Dieudonné determinant $\mathrm{Det}(M)$ of a matrix M with entries in $\mathcal{S}$ has a more canonical interpretation. If $\widetilde{\det}(M)$ is a polynomial representative of $\mathrm{Det}(M)$ in $\mathcal{S}$, its principal symbol $\widetilde{\sigma}(\widetilde{\det}(M))$ will be an element in the commutative graded ring $\mathrm{gr}(\mathcal{S})$. Since multiplication in $\mathrm{gr}(\mathcal{S})$ is commutative, all choices of order for multiplying diagonal elements (from a triangular form of M) will yield the same symbol $\widetilde{\sigma}(\widetilde{\det}(M))$. This means that the "ambiguity" due to commutators in $\mathcal{S}$ vanishes when passing to the principal symbols in $\mathrm{gr}(\mathcal{S})$. Consequently, the resultant, when viewed in terms of its highest-degree components or its image in $\mathrm{gr}(\mathcal{S})$, behaves such as a commutative resultant, providing a well-defined value (e.g. for degree calculations or leading term analysis). This is particularly useful when the properties of interest (such as common solutions related to highest or lowest degree terms) can be derived from $\mathrm{gr}(\mathcal{S})$.

The situation regarding uniqueness and interpretation can simplify in certain well-behaved non-commutative rings. If $\mathrm{gr}(\mathcal{S})$ is a commutative UFD, then properties of the Dieudonné determinant, and thus the resultant, can be related to determinant computations in this simpler graded ring. For example, if $\widetilde{\det}(M)$ is the Dieudonné determinant, its leading term or symbol $\widetilde{\sigma}(\widetilde{\det}(M))$ in $\mathrm{gr}(\mathcal{S})$ correspond to a standard commutative determinant, making analysis more tractable. The well-definedness of the resultant and its degree can often be clearly established in such structured rings.

4.3.1.2 Hermite Form

The Hermite Normal Form (HNF) offers two significant advantages: firstly, it provides a *unique* canonical representation for the row space of the original matrix; secondly, it can be computed *efficiently* for many important classes of rings [64]. The uniqueness property is particularly valuable in the context of skew resultants. While the Dieudonné determinant (and thus the resultant polynomial computed from it) is inherently defined modulo commutators, transforming the σ-Sylvester matrix (or related matrices in its computation) to its Hermite form can lead to

uniquely defined diagonal entries. If the resultant is derived from these unique monic diagonal entries, the Hermite form can aid in selecting a canonical polynomial representative for the resultant class.

In summary, the Hermite Normal Form contributes to defining a canonical resultant representative by providing a deterministic triangulation of the Sylvester matrix. Since unimodular (elementary) row operations preserve the Dieudonné determinant class, and the HNF is unique for a given matrix under a fixed normalisation convention, the product of the diagonal entries of the HNF yields a canonical and reproducible scalar representative of the skew resultant. By this way, the HNF removes the ambiguity arising from elimination order and effectively fixes a specific representative within the commutator class inherent in the determinant's definition.

4.3.2 Operator Elimination

Many families of special functions and orthogonal polynomials satisfy systems of linear functional equations involving multiple distinct operators, such as differential and shift operators. While these mixed relations fully characterise the function, it is often advantageous to decouple the system to isolate the behaviour with respect to a single operator type. For instance, one may wish to derive a pure differential equation or a pure recurrence relation from a set of mixed differential-difference equations. Operator elimination provides the algebraic mechanism to achieve this. By systematically eliminating specific indeterminates from a system of Ore polynomials, one can uncover these implicit pure relations, thereby transforming a complex mixed system into a more tractable form.

This subsection presents a systematic procedure for reducing such a system of operators. Selected indeterminates (irrelevant to the task at hand) are eliminated, while those of interest are retained. The technique is illustrated using classical parametric families such as the Chebyshev and Legendre polynomials [4].

Example 4.3.1. *The Chebyshev polynomials of the first kind are given by*

$$T_n(t) = \sum_{k=0}^{\lfloor n/2 \rfloor} \binom{n}{2k} (t^2 - 1)^k t^{n-2k},$$

where t is the main variable and n, k are parameters. They satisfy

$$\begin{aligned}(1 - t^2)T'_{n+1}(t) - (n + 1)tT_{n+1}(t) + (n + 1)T_n(t) &= 0,\\ T_{n+2}(t) - 2tT_{n+1}(t) + T_n(t) &= 0,\\ (1 - t^2)T''_n(t) - tT'_n(t) + n^2T_n(t) &= 0.\end{aligned}$$

These relations translate naturally into differential (D_t) and shift (S_n) operators in an Ore algebra such as $\mathbb{Q}[n, t][D_t; 1, D_t][S_n; \sigma_n, 0]$, where $D_t(T_n) = T'_n$ and $S_n(T_n) = T_{n+1}$.

(i) Mixed differential–shift operator

$$(1 - t^2)D_tS_n - (n + 1)tS_n - (n + 1),$$

(ii) Pure shift operator

$$S_n^2 - 2tS_n + 1,$$

(iii) Pure differential operator

$$(1 - t^2)D_t^2 - tD_t + n^2.$$

Consider the system consisting of (i) and (ii):

$$\begin{cases} F = (1 - t^2)D_tS_n - (n + 1)tS_n - (n + 1),\\ G = S_n^2 - 2tS_n + 1. \end{cases} \tag{4.8}$$

Eliminating S_n from F and G recovers relation (iii). Conversely, eliminating D_t from (i) and (iii) yields a pure S_n relation.

Other orthogonal polynomials behave similarly. For instance, the Legendre polynomials

$$P_n(t) = \frac{1}{2^n} \sum_{k=0}^{n} \binom{n}{k}^2 (t-1)^{n-k}(t+1)^k$$

satisfy

$$\begin{aligned}(1-t^2)D_t + (n+1)S_n - (n+1)t &= 0,\\ (n+2)S_n^2 - (2n+3)tS_n + (n+1) &= 0,\\ (t^2-1)D_t^2 + 2tD_t - n(n+1) &= 0.\end{aligned}$$

Further examples, such as the binomial coefficient identities explored in Example 4.3.5, also involve multiple parameters and operator types. For more details on such special functions, see for example [4].

A method for simplifying an operator system by eliminating specified indeterminates is therefore highly advantageous. The resultant serves as a powerful tool for this purpose. The following sections will establish a key theorem, with its proof, demonstrating that the resultant of an operator system annihilates any common solution of that system.

Before stating and proving this main theorem, it is necessary to establish a foundational property that connects the resultant to the original polynomials.

In the commutative setting, this property is the well-known Bézout-style identity, wherein the resultant can be expressed as a linear combination of the input polynomials, with polynomial coefficients. Conventional proofs of this identity (e.g. [2, 106]) rely on matrix representations and the use of Cramer's rule. However, this approach is not directly applicable in the present context, since the Dieudonné determinant does not, in general, admit cofactor expansion. For example, consider a 2×2 matrix

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

over a skew field. A standard cofactor expansion yields $ad - bc$, whereas the

Dieudonné determinant is given by $ad - aca^{-1}b$; the two expressions are generally distinct. An alternative approach, namely the permutation-based definition of the determinant, is also inadequate as the non-commutative nature of multiplication makes the order of terms in permutation products ambiguous.

Instead, a method adapted from [40, Proposition 5, p. 164] is used. Although originally developed for the commutative case, this approach does not rely on cofactor expansions or permutation arguments. Rather, it uses a specific matrix identity in conjunction with a non-commutative version of Cramer's rule to equate coefficients. With appropriate modifications, the method extends to the non-commutative setting, where computations must account for the ring's commutation rules and respect the properties of the Dieudonné determinant, namely its definition modulo commutators and its representability as a polynomial, as established in Lemma 3.6.9. This adaptation demonstrated in the proof of the following proposition.

Proposition 4.3.2. *Let $f = \sum_{i=0}^{m} a_i(\theta_1)\theta_2^i$ and $g = \sum_{j=0}^{k} b_j(\theta_1)\theta_2^j$ be bivariate skew polynomials in $\mathcal{S} = \mathcal{F}[\theta_1;\sigma_1][\theta_2;\sigma_2]$, of respective degrees $m, k > 0$ in θ_2. Then there exist polynomials $p, q \in \mathcal{S}$ with coefficients in $\mathcal{F}[\theta_1;\sigma_1]$ such that*

$$pf + qg = \mathrm{Res}_{\theta_2}(f, g). \tag{4.9}$$

Proof. The proof begins by aiming to find polynomials $P = \sum_{i=0}^{k-1} p_i(\theta_1)\theta_2^i$ and $Q = \sum_{j=0}^{m-1} q_j(\theta_1)\theta_2^j$ that satisfy the identity

$$Pf + Qg = 1. \tag{4.10}$$

Expanding this equation and collecting terms by powers of θ_2 (from θ_2^0 to θ_2^{m+k-1}) leads to a system of $m + k$ linear equations for the $m + k$ unknown coefficients

$p_i(\theta_1)$ and $q_j(\theta_1)$. Letting $c^{[i]} = \sigma_2^i(c)$, this system has the form:

$$\begin{aligned}
&p_{k-1}a_m^{[k-1]} + q_{m-1}b_k^{[m-1]} = 0,\\
&p_{k-1}a_{m-1}^{[k-1]} + p_{k-2}a_m^{[k-2]} + q_{m-1}b_{k-1}^{[m-1]} + q_{m-2}b_k^{[m-2]} = 0,\\
&\qquad\vdots \qquad\qquad \vdots\\
&p_1a_0^{[1]} + p_0a_1^{[0]} + q_1b_0^{[1]} + q_0b_1^{[0]} = 0,\\
&p_0a_0^{[0]} + q_0b_0^{[0]} = 1.
\end{aligned}$$

This system can be written in matrix form as:

$$\Big(p_{k-1},\ \cdots,\ p_0,\ q_{m-1},\ \cdots,\ q_0\Big)\begin{pmatrix}
a_m^{[k-1]} & a_{m-1}^{[k-1]} & \cdots & a_0^{[k-1]} & & \\
 & a_m^{[k-2]} & \cdots & a_1^{[k-2]} & a_0^{[k-2]} & \\
 & & \ddots & & \ddots & \\
 & & & a_m^{[0]} & \cdots & a_0^{[0]}\\
b_k^{[m-1]} & b_{k-1}^{[m-1]} & \cdots & b_0^{[m-1]} & & \\
 & b_k^{[m-2]} & \cdots & b_1^{[m-2]} & b_0^{[m-2]} & \\
 & & \ddots & & \ddots & \\
 & & & b_k^{[0]} & \cdots & b_0^{[0]}
\end{pmatrix} = \Big(0,\ \cdots,\ 0,\ 1\Big).$$

or simply

$$\Big(p_{k-1},\ \ \ldots,\ \ p_0,\ \ q_{m-1},\ \ \ldots,\ \ q_0\Big)\cdot\mathbf{M} = \Big(0,\ \ \ldots,\ \ 0,\ \ 1\Big),$$

where $\mathbf{M}$ is the $(m+k)\times(m+k)$ σ-Sylvester matrix of f and g. The Dieudonné determinant of $\mathbf{M}$ is, by definition, the resultant $\mathrm{Res}_{\theta_2}(f,g)$.

Assuming $\mathrm{Res}_{\theta_2}(f,g)\neq 0$, the σ-Sylvester matrix $\mathbf{M}$ is invertible over the skew field $\mathcal{F}(\theta_1;\sigma_1)$, and a non-commutative version of Cramer's rule can be applied.

For instance, the coefficient q_0 is given by an expression of the form:

$$q_0 \cdot \mathrm{Res}_{\theta_2}(f,g) = \mathrm{Det}\begin{pmatrix} a_m^{[k-1]} & \cdots & a_0^{[k-1]} & & & \\ & a_m^{[k-2]} & \cdots & a_0^{[k-2]} & & \\ & & \ddots & & \ddots & \\ & & & a_m^{[0]} & \cdots & a_0^{[0]} \\ b_k^{[m-1]} & \cdots & b_0^{[m-1]} & & & \\ & \ddots & & \ddots & & \\ & & b_k^{[1]} & \cdots & b_0^{[1]} & \\ 0 & \cdots & & \cdots & 0 & 1 \end{pmatrix}$$

that is

$$q_0 \cdot \mathrm{Res}_{\theta_2}(f,g) = \mathrm{Det}\begin{pmatrix} \text{first } m+k-1 \text{ rows of } \mathbf{M} \\ 0 \quad \cdots \quad 0 \quad 1 \end{pmatrix}. \tag{4.11}$$

By Lemma 3.6.9, the determinant on the right-hand side, $\mathrm{Det}(\mathbf{M}_{q_0})$, can be represented by a polynomial, which we denote $Q_0(\theta_1) \in \mathcal{F}[\theta_1;\sigma_1]$. Since Dieudonné determinants commute, we can write $q_0 = (\mathrm{Res}_{\theta_2}(f,g))^{-1} \cdot Q_0$. In general, for each i and j:

$$p_i = (\mathrm{Res}_{\theta_2}(f,g))^{-1} \cdot P_i(\theta_1) \quad \text{and} \quad q_j = (\mathrm{Res}_{\theta_2}(f,g))^{-1} \cdot Q_j(\theta_1),$$

for some polynomials $P_i, Q_j \in \mathcal{F}[\theta_1;\sigma_1]$.

Let the polynomials $P_{res}, Q_{res} \in \mathcal{S}$ be defined by $P_{res} = \sum_{i=0}^{k-1} P_i\theta_2^i$ and $Q_{res} = \sum_{j=0}^{m-1} Q_j\theta_2^j$. Then the original solution polynomials P and Q can be expressed as:

$$P = (\mathrm{Res}_{\theta_2}(f,g))^{-1} \cdot P_{res} \quad \text{and} \quad Q = (\mathrm{Res}_{\theta_2}(f,g))^{-1} \cdot Q_{res}.$$

Substituting these expressions back into Equation (4.10) and clearing the denom-

inator by left-multiplying by $\mathrm{Res}_{\theta_2}(f, g)$ yields the desired identity:

$$P_{res} f + Q_{res} g = \mathrm{Res}_{\theta_2}(f, g).$$

Renaming P_{res} and Q_{res} as p and q completes the proof. □

Before proceeding to the main theorem in this section, the definition of an annihilating ideal is recalled.

Definition 4.3.3. *Let $\mathcal{V}$ be an algebra of functions and let $\mathbf{u} \in \mathcal{V}$. Let $\mathcal{S}$ be a ring of operators acting on $\mathcal{V}$. The* annihilating ideal *of $\mathbf{u}$ in $\mathcal{S}$, denoted* $\mathrm{Ann}(\mathbf{u})$*, consists of all operators $f \in \mathcal{S}$ that annihilate $\mathbf{u}$:*

$$\mathrm{Ann}(\mathbf{u}) = \{f \in \mathcal{S} \mid f \cdot \mathbf{u} = 0\}. \tag{4.12}$$

Now the main theorem can be proved which states that the resultant of a system of operators belongs to the annihilating ideal of any common solution of the system.

Theorem 4.3.4. *The resultant of a system of two bivariate skew polynomial operators annihilates any common solution of the system.*

Proof. Let $r = \mathrm{Res}_{\theta_2}(f, g)$ be the resultant of two operators $f, g \in \mathcal{S}$. Proposition 4.3.2 ensures the existence of operators $p, q \in \mathcal{S}$ such that

$$pf + qg = r. \tag{4.13}$$

Let $\mathbf{u}$ be a common solution of the system, such that $f \cdot \mathbf{u} = 0$ and $g \cdot \mathbf{u} = 0$.

Applying the operator identity (4.13) to $\mathbf{u}$ yields:

$$\begin{aligned} r \cdot \mathbf{u} &= (pf + qg) \cdot \mathbf{u} \\ &= (pf) \cdot \mathbf{u} + (qg) \cdot \mathbf{u} \\ &= 0 + 0 \quad (\text{since } f \cdot \mathbf{u} = 0 \text{ and } g \cdot \mathbf{u} = 0) \\ &= 0. \end{aligned}$$

Therefore, $r = \mathrm{Res}_{\theta_2}(f, g)$ belongs to $\mathrm{Ann}(\mathbf{u})$. □

The following example illustrates the theorem.

Example 4.3.5. *Consider the binomial coefficient* $\mathrm{C}(n,k) = \binom{n}{k}$*, which satisfies both Pascal's identity and a contiguous relation:*

$$\binom{n+1}{k+1} - \binom{n}{k+1} - \binom{n}{k} = 0, \tag{4.14}$$

$$(k+1)\binom{n}{k+1} - (n-k)\binom{n}{k} = 0. \tag{4.15}$$

These relations can be reformulated using shift operators S_n *and* S_k*, where*

$$S_n \cdot \mathrm{C}(n,k) = \mathrm{C}(n+1,k) \text{ and } S_k \cdot \mathrm{C}(n,k) = \mathrm{C}(n,k+1),$$

to form the operator system:

$$\begin{cases} F &= S_n S_k - S_k - 1, \\ G &= (k+1)S_k - (n-k). \end{cases} \tag{4.16}$$

To derive a new relation involving only S_n*, the resultant can be used to eliminate* S_k*. For this purpose, the system is interpreted as formed by bivariate skew polynomials in* $\mathbb{Q}(n,k)[\theta_1;\sigma_1][\theta_2;\sigma_2]$*, where the indeterminates* θ_1 *and* θ_2 *correspond to the operators* S_n *and* S_k*, respectively. Here,* σ_1 *denotes the shift operator with respect to* n*, and* σ_2 *with respect to* k*. The system then becomes:*

$$\begin{cases} f &= (\theta_1 - 1)\theta_2 - 1, \\ g &= (k+1)\theta_2 - (n-k). \end{cases} \tag{4.17}$$

Note that $\deg_{\theta_2}(f) = 1$ *and* $\deg_{\theta_2}(g) = 1$. *The computation of* $\mathrm{Res}_{\theta_2}(f, g)$ *using the Dieudonné determinant of the* 2×2 σ*-Sylvester matrix yields:*

$$\mathrm{Res}_{\theta_2}(f, g) = \mathrm{Det} \begin{pmatrix} \theta_1 - 1 & -1 \\ k+1 & -(n-k) \end{pmatrix}$$

Applying elementary row operations, the determinant can be rewritten as:

$$= \mathrm{Det} \begin{pmatrix} k+1 & -(n-k) \\ 0 & (k+1)^{-1}(\theta_1 - 1)(n-k) - 1 \end{pmatrix}$$

$$\begin{aligned} &= (k+1)\left((k+1)^{-1}(\theta_1 - 1)(n-k) - 1\right) \\ &= (\theta_1 - 1)(n-k) - (k+1) \\ &= \theta_1 n - \theta_1 k - n - 1 \\ &= \sigma_1(n)\theta_1 - \sigma_1(k)\theta_1 - n - 1 \\ &= (n+1)\theta_1 - k\,\theta_1 - n - 1 \\ &= (n-k+1)\theta_1 - (n+1). \end{aligned} \tag{4.18}$$

By Theorem 4.3.4, this resultant must annihilate $\binom{n}{k}$. *Translating back into operator notation gives the relation:*

$$(n-k+1)S_n - (n+1), \tag{4.19}$$

which is a pure S_n*-based operator that annihilates* $\mathrm{C}(n, k)$. *Its correctness can be directly verified by applying it to the binomial coefficient* $\binom{n}{k}$.

Thus far, the discussion has focused primarily on the bivariate case. Extending this framework to the trivariate (and ultimately to the general multivariate)

setting introduces substantial complications. In particular, the corresponding coefficient matrix would involve entries that are polynomials in two or more indeterminates. This new coefficient ring is no longer a Euclidean domain (nor a Principal Ideal Domain), which is the necessary property for the direct triangularisation method. As a result, the direct method outlined above becomes inapplicable.

To address this challenge, an alternative (and more efficient) approach is introduced, based on a suitable non-commutative evaluation and interpolation technique. This method will be developed in detail in the next chapter.

Chapter Summary

This chapter establishes a structured elimination theory for bivariate skew (Ore) polynomial systems. Building on classical resultant theory, it overcomes the challenges inherent in non-commutative algebra (such as the lack of a unique determinant and the failure of standard Bézout-style identities) by defining an effective skew resultant in Ore algebra. The primary goal is to develop a foundational tool for both theoretical analysis and practical applications, including operator identities and cryptographic constructions.

The skew resultant $\mathrm{Res}_{\theta_2}(f, g)$ is formally defined as the Dieudonné determinant of the σ-Sylvester matrix associated with two polynomials $f(\theta_1, \theta_2)$ and $g(\theta_1, \theta_2)$ from an iterated Ore ring. The construction of this matrix is motivated by two equivalent algebraic conditions; the existence of a non-trivial Bézout-like relation ($Pf + Qg = 0$) and that of a least common left multiple ($Uf = Vg$). Although the computation occurs over the skew field of fractions $\mathcal{F}(\theta_1; \sigma_1)$, the resultant is shown to be representable as a polynomial in $\mathcal{F}[\theta_1; \sigma_1]$, a key property that makes it both adaptable and useful for practical applications.

To compute this resultant directly, Algorithm 3 is presented. This method generalises Gaussian elimination, triangularising the σ-Sylvester matrix via unimodular row operations within the non-commutative ring of coefficients. The

resultant is then the product of the diagonal entries. Further structural properties are also explored; for instance, in almost-commutative rings, the resultant's leading symbol behaves like a classical commutative resultant, and canonical representatives can be obtained via Hermite forms.

Crucially, the resultant is established as a valid elimination tool. A non-commutative analogue of the Bézout identity $pf + qg = \mathrm{Res}_{\theta_2}(f, g)$ is proven (Proposition 4.3.2), which directly implies the main theoretical result (Theorem 4.3.4); the resultant annihilates any common solution of an operator system. The practical significance of this result is confirmed through its application to operator identities for Chebyshev polynomials and binomial coefficients (Example 4.3.5). However, the direct triangularisation approach becomes computationally infeasible for systems with three or more indeterminates. This limitation motivates the development of the more scalable evaluation-interpolation framework presented in the subsequent chapter.

Ultimately, this chapter provides a self-contained theoretical and algorithmic foundation for bivariate skew resultants, establishing their crucial properties as elimination tools. This work serves as the necessary algebraic foundation for the advanced computational methods and novel cryptographic applications.

Chapter 5

Efficient Computation of Skew Polynomial Resultants via Evaluation and Interpolation

While the preceding chapter laid the theoretical groundwork for skew polynomial resultants, their direct computation via the Dieudonné determinant of large σ-Sylvester matrices can be computationally prohibitive, particularly as the degrees of the input polynomials grow. This chapter, therefore, develops an alternative and more computationally efficient methodology based on an evaluation-interpolation framework.

This modular strategy works by reducing a bivariate problem into multiple, simpler univariate instances. Specifically, this is achieved by evaluating (via operator evaluation) one of the indeterminates at a sequence of distinct points. The resultants of these univariate instances, which lie in the base field, are then computed. Finally, the bivariate resultant is reconstructed from these scalar resultants using a suitable interpolation technique.

The sections that follow detail the theoretical foundation for non-commutative evaluation and interpolation in this setting, address the specific challenges encountered in Ore polynomial rings, present the complete algorithm, and demon-

strate its effectiveness through illustrative examples.

5.1 Efficient Computation via Evaluation and Interpolation

This section presents an alternative method for computing the resultant of matrices with skew polynomial entries, based on evaluation and interpolation techniques. The approach proceeds by first identifying a suitable evaluation map from the various non-commutative formulations available. A key theorem is then stated, establishing the connection between bivariate resultants and evaluation maps, and its essential role in the elimination process is highlighted. As a consequence, an evaluation–interpolation method is introduced that extends the commutative case developed in [112].

Both the evaluation and interpolation stages introduce significant challenges. These arises from the fundamental differences between evaluation maps in the skew and commutative settings, as well as from the requirement to preserve the non-commutative product rule during computations. In the commutative case, as demonstrated in methods such as [112], input polynomials can be evaluated at scalar values, and the computations remain consistent throughout. In contrast, applying scalar evaluation directly to skew polynomials may lead to inconsistencies.

For instance, consider the operator expression θt, where θ denotes the differential operator D with respect to the variable t. This expression is equivalent to $t\theta + 1$ (due to the operator commutation rule $D\mathbf{u} = \mathbf{u}D + \frac{d}{dt}\mathbf{u}$). If one attempts to evaluate $t\theta + 1$ by naively substituting a scalar value for θ, say $\theta = 2$, results in $t(2) + 1 = 2t + 1$. However, applying the same naive substitution to the original expression θt yields $2t$. The discrepancy between $2t + 1$ and $2t$ illustrates that simple scalar substitution is not a valid ring homomorphism and fails to preserve

the underlying algebraic structure in this non-commutative case.

This inconsistency highlights the importance of selecting a valid evaluation map from among the various formulations proposed in the literature, such as the remainder theorem [32], the product formula ([32], [Lemma 8.6.4]), operator evaluation [10], and the recursive relation framework ([87], [§2]). The choice of evaluation map is critical and must be made with due regard to the specific algebraic context under consideration, as discussed in the remainder of this section.

Summary and Conceptual Flow

The main ideas of this section can be summarised as follows.

- The direct Dieudonné-determinant approach to skew resultants is conceptually natural but computationally expensive, since it operates on large σ-Sylvester matrices with entries in a skew polynomial ring.
- The evaluation–interpolation strategy reduces this cost by converting a single bivariate elimination problem into a family of simpler univariate (or scalar) problems obtained by evaluating one indeterminate at carefully chosen operator or scalar points.
- In the skew setting, naive scalar substitution does *not* define a ring homomorphism and can destroy the non-commutative product rule. A central task of this section is therefore to select an evaluation map that is compatible with the Ore structure (e.g. operator evaluation or related constructions in the literature).
- Once a valid evaluation map is fixed, each evaluation instance yields a smaller resultant computation (typically over the base field or a simpler skew ring). These scalar or univariate resultants are then used as interpolation data to reconstruct the original bivariate skew resultant.
- The correctness of this procedure relies on a key theorem linking the vanishing behaviour of the evaluated resultants to the vanishing of the original

skew resultant, thereby ensuring that elimination is preserved under evaluation and interpolation.

The overall workflow of the method can be viewed schematically as:

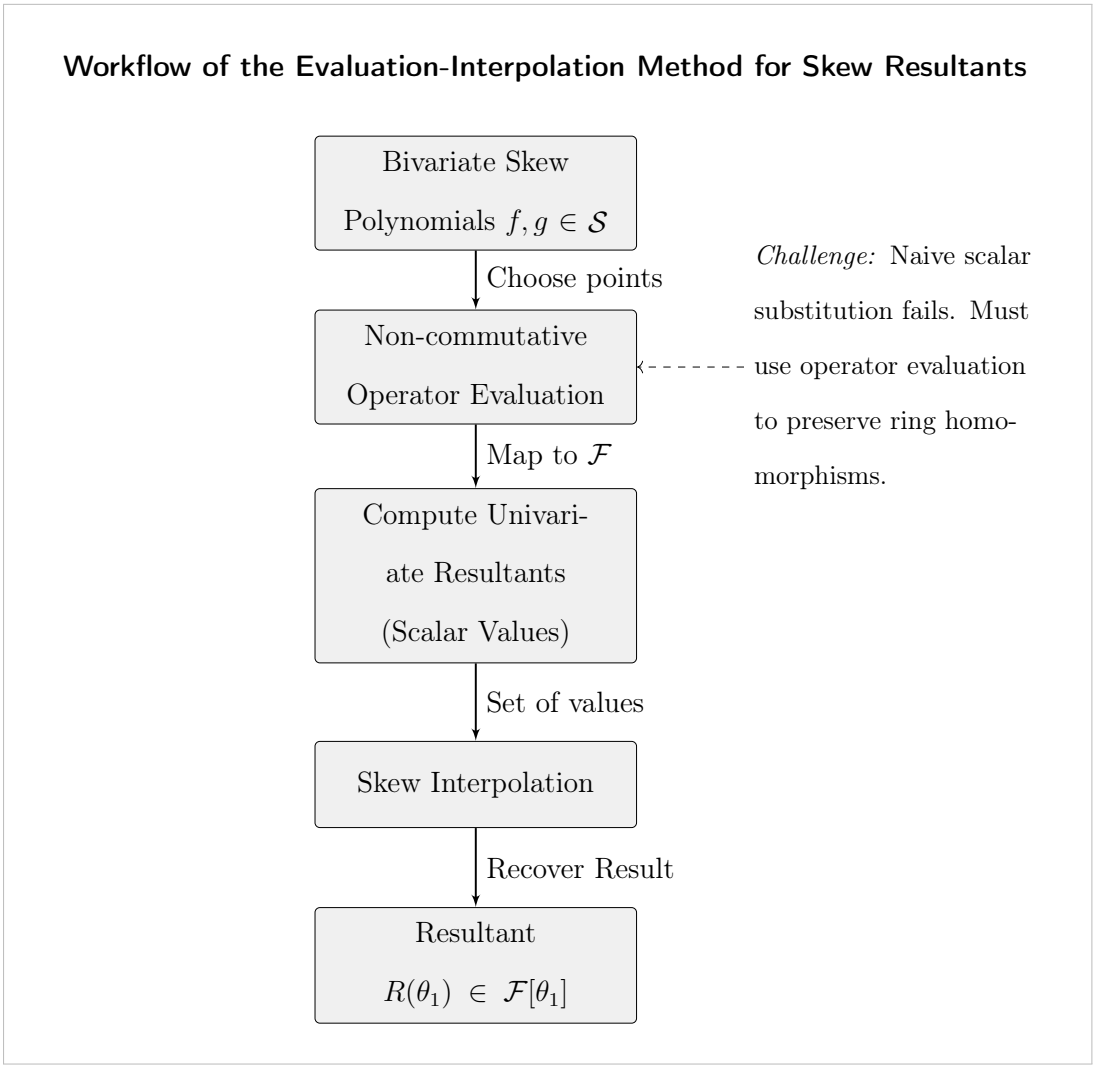

Figure 5.1: Processing pathway for the evaluation-interpolation method.

This summary is intended to provide a conceptual guide through the technical details that follow: the subsequent subsections develop the precise evaluation maps, state and prove the key linking theorem, and formalise the interpolation procedure into an explicit algorithm.

5.1.1 Skew Polynomial Evaluation

A key distinction between evaluation in commutative and non-commutative rings lies in the fact that, in the latter, evaluation maps are not generally ring homomorphisms and therefore do not preserve multiplicative structure, as demonstrated in the preceding example. In this study, a ring homomorphic evaluation map is pursued, one that preserves the algebraic structure required for the proposed computational methods.

Since the elements of Ore polynomial rings can be naturally interpreted as operators, it is intuitive to consider an evaluation map that operates on such entities. A particularly suitable choice is to evaluate at one of the ring's automorphisms, such as σ_1 or σ_2, especially when the chosen map is known to act as a ring homomorphism, an essential property for the evaluation process to maintain algebraic consistency. Accordingly, the operator evaluation framework from [10] is adapted here, with refinements introduced to accommodate the bivariate setting.

In addition to structural benefits, operator evaluation techniques can also enhance the computational efficiency of polynomial multiplication during the computation of resultants, as evidenced by studies such as [22, 63]. The approach adopted here differs slightly: one of the automorphisms, namely σ_1, is interpreted as an element of an appropriately extended base field. For convenience, the notation $\tilde{\mathcal{F}}$ is introduced to denote the field of rational functions in the operator variable, for instance, $\mathcal{F}(\sigma_1, \circ)$. This formalism offers a convenient setting for the elimination techniques developed herein.

The following definition formalises the concept of operator evaluation in the bivariate skew polynomial setting.

Definition 5.1.1 (Also see [113]). *Let $\tilde{\mathcal{F}}[\theta_1;\sigma_1][\theta_2;\sigma_2]$ denote a bivariate skew polynomial ring. This ring may be viewed as $\mathcal{S} = E[\theta_1;\sigma_1]$, where the coefficient ring is $E = \tilde{\mathcal{F}}[\theta_2;\sigma_2]$. For any polynomial $f = \sum_{i=0}^{n} \alpha_i \theta_1^i$ with $\alpha_i \in E$, the*

evaluation map $\text{eval}_{(\theta_1\text{-}\sigma_1)}(f)$ *is defined as*

$$\text{eval}_{(\theta_1\text{-}\sigma_1)} : \begin{array}{c} E[\theta_1;\sigma_1] \to E \\ f = \sum_{i=0}^{n} \alpha_i \theta_1^i \mapsto f(\theta_2, \sigma_1) = \sum_{i=0}^{n} \alpha_i \sigma_1^i. \end{array}$$

The map $\text{eval}_{(\theta_1\text{-}\sigma_1)}$ satisfies the homomorphism property, a critical aspect confirmed in the following result.

Lemma 5.1.2 (Also see [113])**.** *The map* $\text{eval}_{(\theta_1\text{-}\sigma_1)}$ *is a ring homomorphism.*

Proof. Let $f = \sum_{i=0}^{n} \alpha_i \theta_1^i$ and $g = \sum_{j=0}^{m} \beta_j \theta_1^j$ be two polynomials in the skew-polynomial ring $\mathcal{S} = (\tilde{\mathcal{F}}[\theta_2;\sigma_2])[\theta_1;\sigma_1]$, where the coefficients α_i, β_j are in $\tilde{\mathcal{F}}[\theta_2;\sigma_2]$.

The homomorphism property holds for addition and the multiplicative identity. To show it preserves multiplication, recall that the product fg in $\mathcal{S}$ is given by $fg = \sum_{i,j} \alpha_i \sigma_1^i(\beta_j)\theta_1^{i+j}$. Applying the evaluation map yields:

$$\begin{aligned} \text{eval}_{(\theta_1\text{-}\sigma_1)}(fg) &= \text{eval}_{(\theta_1\text{-}\sigma_1)} \left(\sum_{i=0}^{n} \sum_{j=0}^{m} \alpha_i \sigma_1^i(\beta_j) \theta_1^{i+j} \right) \\ &= \sum_{i=0}^{n} \sum_{j=0}^{m} \alpha_i \sigma_1^i(\beta_j) \sigma_1^{i+j} \\ &= \left(\sum_{i=0}^{n} \alpha_i \sigma_1^i \right) \left(\sum_{j=0}^{m} \beta_j \sigma_1^j \right) \\ &= \text{eval}_{(\theta_1\text{-}\sigma_1)}(f) \text{eval}_{(\theta_1\text{-}\sigma_1)}(g). \end{aligned}$$

□

The evaluation map can also be extended to the abelianised multiplicative group of a skew field (i.e. the multiplicative group modulo its commutator subgroup) as defined below.

Definition 5.1.3 (Also see [113])**.** *Let* $\tilde{\mathcal{F}}(\theta;\sigma)$ *be a skew field, and let* $\tilde{\mathcal{F}}^{\times}(\theta;\sigma)$ *denote its multiplicative group. The modular evaluation map* $\text{eval}_{(\theta\text{-}\sigma)}$ *is defined*

on the abelianised group as follows:

$$\text{eval}_{(\theta\text{-}\sigma)} : \begin{array}{l} \tilde{\mathcal{F}}^{\times}(\theta;\sigma)/[\tilde{\mathcal{F}}^{\times}(\theta;\sigma), \tilde{\mathcal{F}}^{\times}(\theta;\sigma)] \to \tilde{\mathcal{F}}^{\times}/[\tilde{\mathcal{F}}^{\times}, \tilde{\mathcal{F}}^{\times}] \\ f = g^{-1}h \mod \mathcal{C} \mapsto f(\sigma) = (g(\sigma))^{-1}h(\sigma) \mod \mathcal{C}', \text{ given } g(\sigma) \neq 0. \end{array}$$

In particular, if $f = \sum_{i=0}^{n} a_i\theta^i$ *is an Ore polynomial, then*

$$\text{eval}_{(\theta\text{-}\sigma)} \colon f = \sum_{i=0}^{n} a_i\theta^i \mod \mathcal{C} \mapsto f(\sigma) = \sum_{i=0}^{n} a_i\sigma^i \mod \mathcal{C}',$$

where $\mathcal{C} = [\tilde{\mathcal{F}}^{\times}(\theta;\sigma), \tilde{\mathcal{F}}^{\times}(\theta;\sigma)]$ *and* $\mathcal{C}' = [\tilde{\mathcal{F}}^{\times}, \tilde{\mathcal{F}}^{\times}]$.

Remark 5.1.4. *Throughout this study, the evaluation map* eval *is, by default, understood to be the modular evaluation defined in Definition 5.1.3 when the argument lies in a quotient modulo commutators.*

Summary of Key Ideas

- In skew (Ore) polynomial rings, naive scalar substitution is *not* a ring homomorphism and does not in general preserve products. This contrasts with the commutative case and can lead to inconsistent evaluations.
- For the purposes of resultant computation, it is therefore essential to work with a *ring-homomorphic* evaluation map that respects the non-commutative multiplication rules.
- The chosen approach interprets skew polynomials as operators and evaluates them at a ring automorphism (here σ_1), viewed as an element of a suitably extended base field $\tilde{\mathcal{F}}$ of rational functions in the operator variable.
- Definition 5.1.1 introduces the operator evaluation map

$$\text{eval}_{(\theta_1\text{-}\sigma_1)} : E[\theta_1;\sigma_1] \longrightarrow E,$$

which sends θ_1^i to σ_1^i and is adapted to the bivariate skew setting $E =$

$\tilde{\mathcal{F}}[\theta_2; \sigma_2]$.

- Lemma 5.1.2 establishes that this map is a ring homomorphism, ensuring that products of skew polynomials are evaluated consistently. This homomorphic property is crucial for transporting multiplicative information (e.g. determinant factors) to the evaluation domain.

- Definition 5.1.3 extends the evaluation map to the abelianised multiplicative group of the skew field. This extension is the mechanism by which Dieudonné determinants of matrices with skew polynomial entries can be evaluated in a way compatible with the non-commutative structure.

- Remark 5.1.4 specifies that, whenever elements are considered modulo commutators, the default convention is to interpret eval as the modular evaluation of Definition 5.1.3, thereby fixing a consistent usage for later chapters.

- Together, these constructions provide the algebraic foundation required for the evaluation–interpolation framework developed later in this chapter, where skew resultants are computed by evaluating at operator points and reconstructing via interpolation.

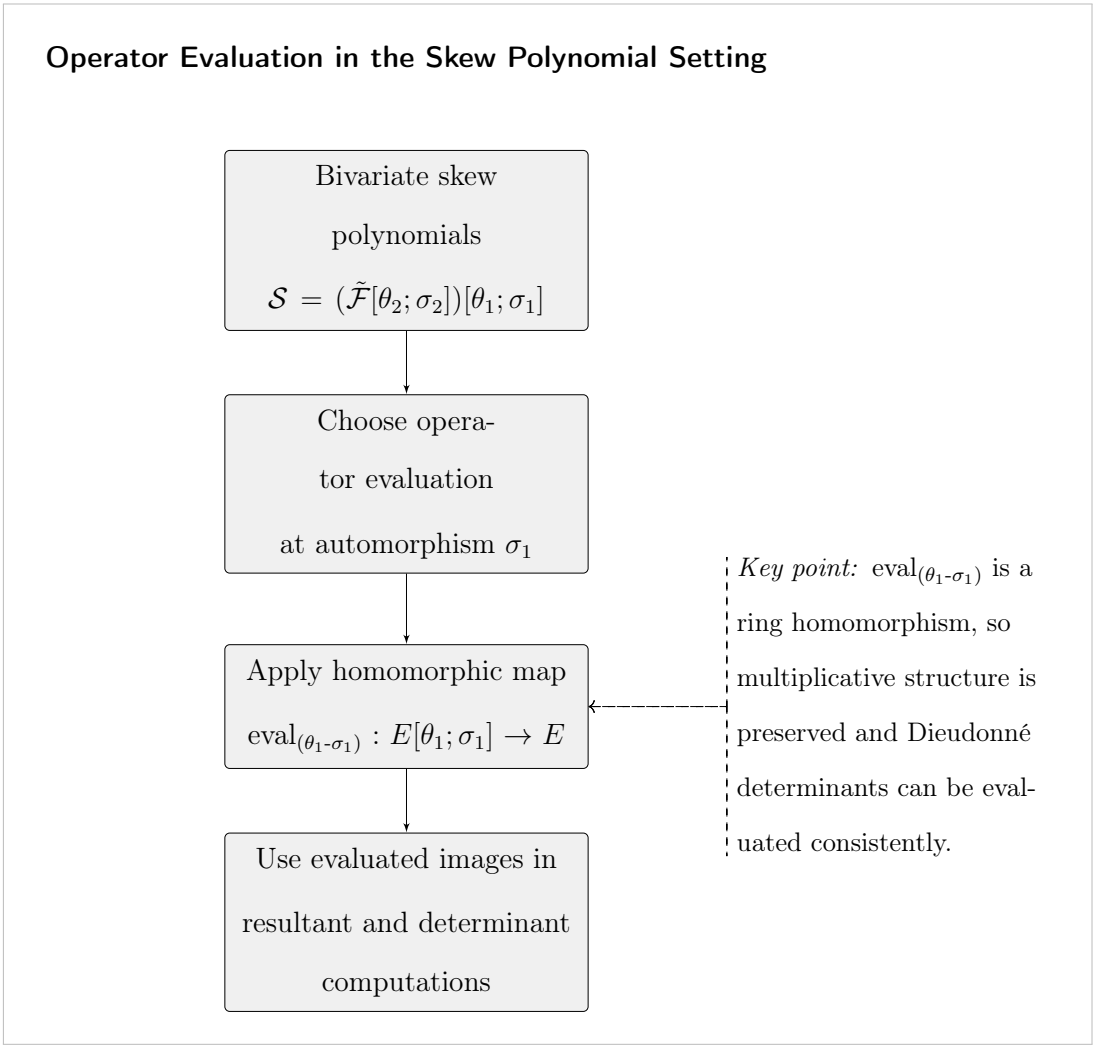

Figure 5.2: Processing pathway for operator evaluation in the skew polynomial setting.

Lemma 5.1.5 (Also see [113])**.** *The evaluation map* $\mathrm{eval}_{(\theta\text{-}\sigma)}$ *as in Definition 5.1.3 is well-defined.*

Proof. Let $\mathcal{C} = [\tilde{\mathcal{F}}^{\times}(\theta;\sigma), \tilde{\mathcal{F}}^{\times}(\theta;\sigma)]$ and $\mathcal{C}' = [\tilde{\mathcal{F}}^{\times}, \tilde{\mathcal{F}}^{\times}]$. The codomain of the map $\mathrm{eval}_{(\theta\text{-}\sigma)}$ is the quotient group $\tilde{\mathcal{F}}^{\times}/\mathcal{C}'$, which is abelian. The proof follows from the universal property of abelianization, as illustrated in the commutative diagram of Figure 5.3.

In the diagram, π is the canonical projection onto the abelianization, and $\varphi_{(\theta\text{-}\sigma)}$ is an auxiliary homomorphism defined as:

$$\varphi_{(\theta\text{-}\sigma)} \colon f \mapsto f(\sigma) \pmod{\mathcal{C}'}, \quad \forall f \in \tilde{\mathcal{F}}^{\times}(\theta;\sigma).$$

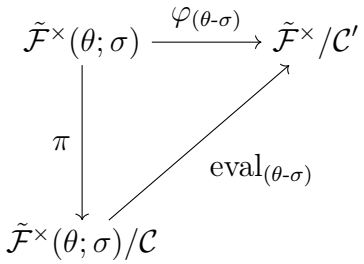

Figure 5.3: Universal property of abelianization.

Since the codomain of $\varphi_{(\theta\text{-}\sigma)}$ is abelian, its kernel must contain the commutator subgroup of the domain, so $\ker(\pi) = \mathcal{C} \leq \ker(\varphi_{(\theta\text{-}\sigma)})$. The universal property guarantees that $\varphi_{(\theta\text{-}\sigma)}$ factors through π, such that $\varphi_{(\theta\text{-}\sigma)} = \text{eval}_{(\theta\text{-}\sigma)} \circ \pi$.

To demonstrate that $\text{eval}_{(\theta\text{-}\sigma)}$ is well-defined, suppose two elements represent the same coset, i.e. $f \mod \mathcal{C} = g \mod \mathcal{C}$. This implies that $fg^{-1} \in \mathcal{C}$. Since $\mathcal{C} \subseteq \ker(\varphi_{(\theta\text{-}\sigma)})$, it follows that $\varphi_{(\theta\text{-}\sigma)}(fg^{-1}) = 1$, which means $\varphi_{(\theta\text{-}\sigma)}(f) = \varphi_{(\theta\text{-}\sigma)}(g)$. From the factorisation, this gives:

$$(\text{eval}_{(\theta\text{-}\sigma)} \circ \pi)(f) = (\text{eval}_{(\theta\text{-}\sigma)} \circ \pi)(g).$$

This is equivalent to $\text{eval}_{(\theta\text{-}\sigma)}(f \mod \mathcal{C}) = \text{eval}_{(\theta\text{-}\sigma)}(g \mod \mathcal{C})$, which confirms the map is well-defined because its value is independent of the choice of representative for the coset. □

The central result of this section, which describes how the resultant behaves under the evaluation map, is now stated.

Theorem 5.1.6 (Also see [113])**.** *Let $\mathcal{S} = \tilde{\mathcal{F}}[\theta_1;\sigma_1][\theta_2;\sigma_2]$. For any polynomials $f, g \in \mathcal{S}$, if the degrees in θ_2 are preserved under evaluation (i.e. if $\deg_{\theta_2}(f) = \deg_{\theta_2}(\text{eval}_{(\theta_1\text{-}\sigma_1)}(f))$ and $\deg_{\theta_2}(g) = \deg_{\theta_2}(\text{eval}_{(\theta_1\text{-}\sigma_1)}(g))$) then the following identity holds:*

$$\text{eval}_{(\theta_1\text{-}\sigma_1)}(\text{Res}_{\theta_2}(f,g)) = \text{Res}_{\theta_2}(\text{eval}_{(\theta_1\text{-}\sigma_1)}(f), \text{eval}_{(\theta_1\text{-}\sigma_1)}(g)). \tag{5.1}$$

Proof. The argument follows from the definition of the resultant and the properties of the ring homomorphism $\text{eval}_{(\theta_1\text{-}\sigma_1)}$. Let $\text{Sylv}_{\theta_2}(f,g)$ denote the $k \times k$

σ-Sylvester matrix.

$$\begin{aligned}
\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(\mathrm{Res}_{\theta_2}(f,g)) &= \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(\mathrm{Det}(\mathrm{Sylv}_{\theta_2}(f,g))) \\
&= \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(\mathrm{Det}(\mathrm{diag}(d_1,\cdots,d_k))), \text{ by Remark 3.6.3} \\
&= \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(\prod_{i=1}^{k} d_i \mod \mathcal{C}) \\
&= \prod_{i=1}^{k} \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(d_i) \mod \mathcal{C}', \text{ as in Definition 5.1.3} \\
&= \mathrm{Det}(\mathrm{diag}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(d_1),\cdots,\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(d_k))) \\
&= \mathrm{Det}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)} \otimes \mathrm{diag}(d_1,\cdots,d_k)) \\
&= \mathrm{Det}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)} \otimes \mathrm{Sylv}_{\theta_2}(f,g)) \\
&= \mathrm{Det}(\mathrm{Sylv}_{\theta_2}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(f),\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(g))) \\
&= \mathrm{Res}_{\theta_2}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(f),\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(g)).
\end{aligned}$$

□

Note that the final two steps of the proof are valid due to the theorem's explicit condition that the polynomial degrees are preserved under evaluation.

Theorem 5.1.6 states that evaluating the resultant is equivalent to computing the resultant of the evaluations. In particular, for any $a \in \mathcal{F}$, one has

$$\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(\mathrm{Res}_{\theta_2}(f,g))(a) = \mathrm{Res}_{\theta_2}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(f),\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(g))(a). \tag{5.2}$$

Here, the left-hand side corresponds to evaluating the resultant computed directly from f and g, whereas the right-hand side is obtained by computing the resultant after evaluating the polynomials themselves. This equivalence enables a reduction to simpler resultant computations over the base ring, which are generally more efficient.

Remark 5.1.7. *An advantage of employing the Dieudonné determinant in The-*

orem 5.1.6 is that the resultant computation reduces to determining a triangular matrix. In the direct method on the left-hand side of Equation (5.2), the diagonal entries are polynomials $d'_i(\theta_1)$. On the right-hand side, their evaluated images $d_i(\sigma_1)$ may be composed to yield

$$\begin{aligned}\mathrm{Res}_{\theta_2}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(f), \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(g))(a) &= \Big(\prod_{i=1}^{k} d_i(\sigma_1)\Big)(a) \\ &= (d_1(\sigma_1)d_2(\sigma_1)\cdots d_k(\sigma_1))(a) \\ &= d_1^*(d_2^*(\cdots d_k^*(a))), \end{aligned} \tag{5.3}$$

where $d_i^* = d_i(\sigma_1)$ for all $i = 1, \ldots, k$.

The evaluation–interpolation framework thus enables the elimination of variables in multivariate skew polynomial systems, a task that is otherwise difficult to carry out directly. Moreover, such techniques are known to yield significant asymptotic improvements in the commutative setting [112], and similar speedups have been observed in skew polynomial arithmetic over finite fields [22, 63]. An analogous improvement is therefore expected in this context. While the approach offers clear efficiency advantages, several challenges must be addressed. The next section outlines the key procedural steps and identifies the associated technical obstacles.

5.1.2 Efficiency Steps and Challenges Encountered

The efficiency of this method is achieved by decomposing the computation into three principal stages, as established by Theorem 5.1.6:

(i) *Evaluation*: Select distinct values a_i $(i = 1, \ldots, k)$, and perform operator evaluation of the polynomials f and g at each a_i with respect to the indeterminate θ_1. This produces a sequence of univariate instances whose coefficients lie in a simplified domain.

(ii) *Partial Resultants*: Compute the univariate resultants of the evaluated polynomials with respect to θ_2.

(iii) *Interpolation*: Reconstruct the full bivariate resultant of f and g from the computed partial resultants using an appropriate non-commutative interpolation technique.

When these steps are applied to skew polynomials, several challenges arise that are not present in the commutative setting:

(i) *Evaluation Points*: Given the variety of available evaluation maps, the result of a chosen evaluation, such as operator evaluation at σ_1 followed by substitution at a value a, may not align with the expectations of standard interpolation methods (e.g., Lagrange or Newton interpolation). A compatible interpolation method must therefore be employed, suited to the specific evaluation scheme used.

(ii) *Validity of the Evaluation Map*: It is essential that the evaluation map preserves ring structure (i.e. it must be a ring homomorphism) to ensure consistency and correctness throughout the evaluation process.

(iii) *Distinct Conjugacy Classes*: For evaluation and interpolation to yield correct results, all chosen evaluation points must lie in pairwise distinct conjugacy classes under the action of the relevant automorphism. This can typically be ensured by selecting primitive elements from a suitable field extension.

(iv) *Unlucky Evaluations*: An evaluation point is termed *unlucky* if it causes the leading coefficient of a polynomial to vanish, thereby lowering the degree of the evaluated polynomial. Such points must be detected and excluded in order to preserve the structural properties required for interpolation.

(v) *Insufficient Evaluation Points*: If the base field does not provide a sufficient number of valid evaluation points (specifically, points that are both lucky

and from distinct conjugacy classes) it becomes necessary to extend the domain by adjoining additional elements from an appropriate field extension.

The subsequent sections examine these issues in greater depth and provide illustrative examples that demonstrate how they may be addressed in practice.

5.1.3 Skew Polynomial Interpolation

This section outlines the interpolation stage of the algorithm, which is formulated with respect to a normal basis. The first subsection introduces the necessary definitions and notational conventions. This is followed by a detailed explanation of the process used to compute the resultant of bivariate skew polynomials through evaluation and interpolation, including techniques for addressing the inherent challenges of this approach. An illustrative example is then provided to demonstrate the methodology in practice.

5.1.3.1 Galois Theory (Finite and Infinite Field Extensions)

To ensure clarity in the discussion that follows, this subsection recalls key definitions and notational conventions from Galois theory that are used throughout the study.

A field $\mathcal{F}$ is said to be a *field extension* of a field $\mathcal{K}$ if $\mathcal{K} \subset \mathcal{F}$, denoted $\mathcal{F}/\mathcal{K}$. Unless otherwise specified, it is assumed that $\mathcal{F}$ is always a field extension of $\mathcal{K}$. A particular case studied in Subsection 5.1.3.3 concerns $\mathcal{K} = \mathbb{Q}$, the field of rational numbers, and $\mathcal{F} = \mathbb{C}$, the field of complex numbers, or a subfield thereof.

Let X be a subset of $\mathcal{F}$; then $\mathcal{K}(X)$ denotes the smallest subfield of $\mathcal{F}$ containing both $\mathcal{K}$ and X. An extension $\mathcal{F}/\mathcal{K}$ is said to be *finitely generated* if there exists a finite set $X \subset \mathcal{F}$ such that $\mathcal{F} = \mathcal{K}(X)$. Furthermore, the extension is *simple* if there exists a single element $\alpha \in \mathcal{F}$ such that $\mathcal{F} = \mathcal{K}(\alpha)$. In this case, one often writes $\mathcal{K}(\alpha)$ instead of $\mathcal{K}(\{\alpha\})$.

An element $\alpha \in \mathcal{F}$ is said to be *algebraic* over $\mathcal{K}$ if there exists a non-zero polynomial m_α over $\mathcal{K}$ such that $m_\alpha(\alpha) = 0$. The *minimal polynomial* of an

algebraic element $\alpha \in \mathcal{F}$ over $\mathcal{K}$ is the unique monic irreducible polynomial m_α over $\mathcal{K}$ satisfying $m_\alpha(\alpha) = 0$, and such that m_α divides any other polynomial over $\mathcal{K}$ having α as a root. That is, m_α has minimal degree among such polynomials.

Viewing $\mathcal{F}$ as a vector space over $\mathcal{K}$, the *degree* of the extension $\mathcal{F}/\mathcal{K}$ is the dimension of this vector space and is denoted by $[\mathcal{F} : \mathcal{K}]$. The extension is said to be *finite* or *finite-dimensional* if $[\mathcal{F} : \mathcal{K}] < \infty$, in which case the dimension is also denoted $\dim_{\mathcal{K}}(\mathcal{F})$.

Now consider the finite extension $\mathcal{K}(\alpha)/\mathcal{K}$ for an algebraic element α over $\mathcal{K}$, defined by

$$\mathcal{F} = \mathcal{K}(\alpha) = \left\{ \sum_{i=0}^{r-1} a_i \alpha^i : a_i \in \mathcal{K} \right\}, \tag{5.4}$$

where $r = \deg(m_\alpha)$ for some minimal polynomial m_α over $\mathcal{K}$. This set forms a finite-dimensional vector space over $\mathcal{K}$ with basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{r-1}\}$.

Let σ be an automorphism of $\mathcal{F}$ that fixes $\mathcal{K}$ (i.e. a $\mathcal{K}$-automorphism). Such a map satisfies

$$\sigma\left(\sum_{i=0}^{r-1} a_i \alpha^i \right) = \sum_{i=0}^{r-1} a_i \sigma(\alpha)^i.$$

It follows that σ is completely determined by the image $\sigma(\alpha)$, which must itself be a root of m_α. This leads to a one-to-one correspondence between the $\mathcal{K}$-automorphisms of $\mathcal{K}(\alpha)$ and the roots of m_α, since each automorphism maps α to another root of the same minimal polynomial. Moreover, this correspondence induces an isomorphism between $\mathcal{K}(\alpha)$ and $\mathcal{K}(\sigma(\alpha))$.

A monic irreducible polynomial over $\mathcal{K}$ is said to be *separable* if it has no repeated roots in any extension of $\mathcal{K}$ (that is all its roots are distinct). An extension $\mathcal{F}/\mathcal{K}$ is called *separable* if every element $\alpha \in \mathcal{F}$ has a separable minimal polynomial over $\mathcal{K}$. The well-known *primitive element theorem* states that every separable finite extension is simple.

An extension $\mathcal{F}/\mathcal{K}$ is said to be *normal* if every irreducible polynomial over $\mathcal{K}$ having at least one root in $\mathcal{F}$ splits completely over $\mathcal{F}$. A finite extension $\mathcal{F}/\mathcal{K}$ is called a *Galois extension* if it is both separable and normal. The *Galois group* of

such an extension is the group of all $\mathcal{K}$-automorphisms of $\mathcal{F}$ under composition, denoted by $\mathrm{Gal}(\mathcal{F}/\mathcal{K})$.

The *Fundamental Theorem of Galois Theory* describes a strong connection between the lattice of intermediate subfields of $\mathcal{F}/\mathcal{K}$ and the subgroup lattice of the Galois group $\mathrm{Gal}(\mathcal{F}/\mathcal{K})$. Specifically, it asserts a one-to-one correspondence between intermediate subfields of $\mathcal{F}$ containing $\mathcal{K}$ and subgroups of $\mathrm{Gal}(\mathcal{F}/\mathcal{K})$.

When extending Galois theory to *infinite field extensions*, where the Galois group $\mathrm{Gal}(\mathcal{F}/\mathcal{K})$ may be infinite, the fundamental theorem requires a more subtle formulation. While the concepts of separability and normality remain applicable, the direct correspondence between subfields and subgroups does not generally hold [39]. To remedy this, a topology (called the *Krull topology*) can be imposed on $\mathrm{Gal}(\mathcal{F}/\mathcal{K})$ [39]. Although the full topological structure is not needed for the present study, it is important to note that the fundamental theorem holds in this setting if attention is restricted to *closed subgroups* of the Galois group. This reformulated correspondence remains valid and serves as the foundation for infinite Galois theory. Further details on the Krull topology can be found in [39].

A *number field* is a finite field extension of $\mathbb{Q}$. An algebraic number whose minimal polynomial has integer coefficients is known as an *algebraic integer*. Notably, every number field can be generated by a single algebraic integer (see, for instance, Theorem 2.2 and Corollary 2.12 in [121]).

A *normal basis* [67] is introduced next. This is a special kind of basis for finite Galois extensions when viewed as vector spaces over the base field. It also admits a generalisation to the infinite case [88], which will be discussed in the next subsection.

5.1.3.2 Normal Basis (Finite and Infinite Cases)

Let $\mathcal{F}/\mathcal{K}$ be a finite Galois extension. The set of *Galois $\mathcal{K}$-conjugates* of an element $\alpha \in \mathcal{F}$ is given by $\{\sigma(\alpha) \mid \sigma \in \mathrm{Gal}(\mathcal{F}/\mathcal{K})\}$. This set encodes the action of the Galois group on α, and is often denoted by $\mathrm{Gal}(\mathcal{F}/\mathcal{K}) \cdot \alpha$, or more simply

$\mathrm{G} \cdot \alpha$, where $G = \mathrm{Gal}(\mathcal{F}/\mathcal{K})$.

An element $\alpha \in \mathcal{F}$ is called *normal* if the set $\mathrm{G} \cdot \alpha = \{\sigma(\alpha) \mid \sigma \in \mathrm{Gal}(\mathcal{F}/\mathcal{K})\}$ forms a basis for $\mathcal{F}$ over $\mathcal{K}$. That is, the conjugates of α under the Galois group are linearly independent and span $\mathcal{F}$. This condition is equivalent to saying that the Galois action places α on a single orbit (see Figure 5.4). A basis of this form is known as a *normal basis* [67], formally defined below.

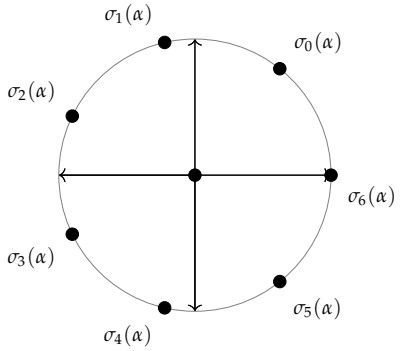

Figure 5.4: Single $\mathrm{Gal}(\mathcal{F}/\mathcal{K})$-orbit of an element $\alpha \in \mathcal{F}$ when $[\mathcal{F} : \mathcal{K}] = 7$.

Definition 5.1.8. *Let $\mathcal{F}/\mathcal{K}$ be a finite Galois extension of degree n with Galois group $\mathrm{Gal}(\mathcal{F}/\mathcal{K}) = \{\sigma_0, \sigma_1, \ldots, \sigma_{n-1}\}$. A basis consisting of all the $\mathcal{K}$-conjugates of an element $\alpha \in \mathcal{F}$, namely*

$$G \cdot \alpha = \{\sigma_0(\alpha), \sigma_1(\alpha), \ldots, \sigma_{n-1}(\alpha)\},$$

is called a normal basis *and is denoted by $\mathcal{N}(\alpha)$.*

A simple example of a normal basis arises from the extension $\mathbb{Q}(i)/\mathbb{Q}$, where the Galois group is $\mathrm{Gal}(\mathbb{Q}(i)/\mathbb{Q}) = \{\sigma_0, \sigma_1\}$, with

$$\sigma_0(i) = i, \quad \sigma_1(i) = -i.$$

Letting $\alpha = 1 + i$, the Galois conjugates are

$$\sigma_0(\alpha) = 1 + i, \quad \sigma_1(\alpha) = 1 - i,$$

and so the normal basis is $\mathcal{N}(\alpha) = \{1 + i, 1 - i\}$.

Note, however, that if $\alpha = i$, then $\{\sigma_0(i), \sigma_1(i)\} = \{i, -i\}$ is not linearly independent over $\mathbb{Q}$, so $\{i, -i\}$ does not form a normal basis.

In the case of an infinite Galois extension $\mathcal{F}/\mathcal{K}$, the definition of a normal basis in Definition 5.1.8 is no longer suitable. This is because the set of Galois conjugates of an element $\alpha \in \mathcal{F}$ is always finite, whereas a $\mathcal{K}$-basis for $\mathcal{F}$ (when infinite-dimensional) must be infinite. To address this, Lenstra [88] proposed a reformulated definition of a normal basis for infinite Galois extensions. The idea is based on a correspondence between $\mathcal{F}$ and the space $C(G, \mathcal{K})$ of all continuous maps from the Galois group G to the field $\mathcal{K}$, where G is equipped with the Krull topology and $\mathcal{K}$ with the discrete topology.

This topological refinement allows the notion of a normal basis to extend naturally to the infinite case, and it recovers the classical definition when $\mathcal{F}/\mathcal{K}$ is finite. Further examples of normal bases may be constructed by adjoining roots of unity to number fields, a topic developed in the next section.

5.1.3.3 Cyclotomic Extension

Cyclotomic fields play a central role in numerous applications involving roots of unity, including representation theory, Kummer theory, and the discrete Fourier transform. In modern cryptography, certain elliptic curves defined over cyclotomic fields are employed in advanced cryptographic schemes. This section provides a brief overview of cyclotomic field extensions, which will be utilised in the next subsection.

It is straightforward to verify that the polynomial $\theta^n - 1$ is separable over $\mathcal{K}$, since it is relatively prime to its derivative $n\theta^{n-1}$. Consequently, $\theta^n - 1$ has n distinct roots in its splitting field over $\mathcal{K}$. In $\mathbb{C}$, these roots are given by

$$\mu_n = \{e^{2\pi ik/n} \mid k = 0, \ldots, n-1\} = \langle \alpha \rangle, \quad \text{where } \alpha = e^{2\pi i/n},$$

which forms a cyclic multiplicative group of order n generated by α. In general, α is not the only generator of μ_n. Any element of μ_n that generates the entire

group is called a *primitive n-th root of unity*. The term *cyclotomic* derives from the Greek for "circle cutting", as the n-th roots of unity divide the unit circle in the complex plane into n equal arcs (see Figure 5.5 for the case $n = 7$).

For any integer k, an element α^k is a primitive n-th root of unity if and only if $\gcd(k, n) = 1$. This follows from the fact that the order of α^k in μ_n is given by $n/\gcd(k, n)$.

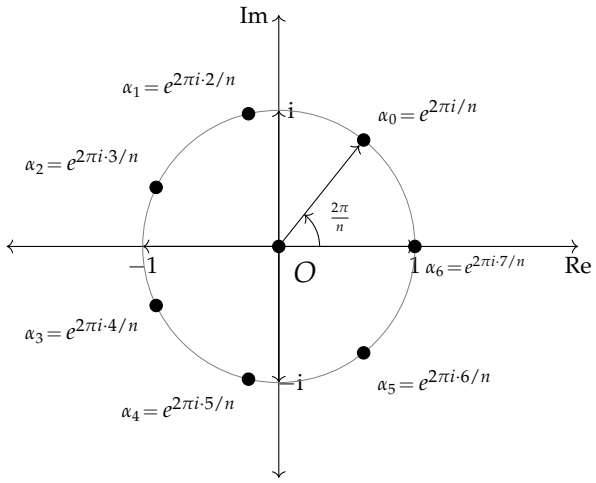

Figure 5.5: Roots of unity for $n = 7$.

Since μ_n is a cyclic group, the map $\alpha \mapsto \alpha^k$ sends a generator to another generator if and only if $\gcd(k, n) = 1$. Therefore, this mapping defines an automorphism of μ_n precisely when this condition holds.

Note that all primitive elements of μ_n are powers of one another. As a result, the extension $\mathcal{K}(\alpha)$ is independent of the particular primitive n-th root α chosen. When $\mathcal{K} = \mathbb{Q}$, the extension $\mathcal{K}(\alpha)$ is called the *n-th cyclotomic field*, consisting of the field generated by all n-th roots of unity over the rationals. In particular, if $n = p$ is prime, a normal basis can be constructed by selecting an initial generator $\alpha_0 = e^{2\pi i/n}$ and applying the Galois action, as illustrated by comparing Figures 5.4 and 5.5.

A notable advantage of cyclotomic extensions is their applicability over arbitrary base fields. One may simply adjoin the n-th roots of unity to $\mathcal{K}$, for a fixed integer $n \neq 0$, to form the extension $\mathcal{K}(\alpha)$. In the case where $\mathcal{K}$ is not of characteristic zero, care must be taken to ensure that n is not divisible by the

characteristic of $\mathcal{K}$. For practical use in modular algorithms, one may select a sufficiently large prime p as the value of n, thereby guaranteeing the availability of a sufficient number of values for the interpolation process, a topic addressed in the following subsection.

5.1.3.4 Non-Commutative Evaluation and Galois Extensions

Non-commutative evaluation differs fundamentally from standard substitution in that naive scalar evaluation fails to preserve multiplicativity, whereas operator evaluation at an automorphism restores the *ring homomorphism* property required for determinant-based elimination. Finite or infinite Galois extensions equipped with a suitable automorphism σ then provide families of evaluation points given by σ-orbits $\{\alpha, \sigma(\alpha), \ldots, \sigma^{m-1}(\alpha)\}$ that are sufficiently long and pairwise distinct to support interpolation. In this setting, the associated skew Vandermonde-type matrices are invertible, ensuring that the interpolation problem is well defined and that the target skew polynomial (such as a resultant) is uniquely determined by its evaluated values.

5.1.3.5 Interpolation

This subsection applies the main theorem to derive an evaluation and interpolation method for computing the resultant of bivariate skew polynomials over number fields, specifically over $\mathbb{Q}(\alpha)$, where α is a complex root of unity. Several examples are provided to illustrate the method.

Recall that the theorem employs operator evaluation to evaluate the polynomials f and g at selected values. However, this evaluation behaves quite differently from standard evaluation maps in the non-commutative setting. For example, consider the operator evaluation map $\mathrm{eval}_{(\theta\text{-}\sigma)(a)}$ applied to $f = \theta^2$ in a skew polynomial ring $\mathcal{F}[\theta; \sigma]$, where $a \in \mathcal{F}$. In this case, θ^2 is interpreted as σ^2, and the evaluation corresponds to applying σ^2 to the value a, i.e. computing $\sigma^2(a)$. This is not equivalent to the typical non-commutative evaluation $\sigma(a) \cdot a$, nor to

the expression $(\sigma(a))^2$, as the latter equals $\sigma(a^2)$, which in general differs from $\sigma^2(a)$. It is also not the same as a naive substitution a^2.

Hence, from an interpolation perspective, the crucial question is: at which value is the indeterminate in the original polynomial truly being evaluated? The answer to this question guides the selection of an appropriate interpolation method. To correctly reconstruct the original polynomial, an interpolation technique must be chosen that is compatible with the specific properties of operator evaluation. This forms the topic of the next section.

5.1.4 Evaluation and Interpolation Technique

This section introduces a modular technique for computing the resultant using an evaluation–interpolation framework, based on the identity established in Theorem 5.1.6. The method employs a coefficient compression strategy: it matches the coefficients on both sides of the identity at sufficiently many evaluation points, and subsequently solves a linear system to determine the unknowns. A similar approach has been successfully applied in [22, 63] for multiplying univariate skew polynomials.

Let $\mathcal{F}/\mathcal{K}$ be a finite Galois extension of degree r. Consider the computation of the resultant of two bivariate skew polynomials f and g in the ring $\mathcal{F}[\theta_1;\sigma_1][\theta_2;\sigma_2]$. A specific case of interest arises when $\mathcal{K} = \mathbb{Q}$ and $\mathcal{F} = \mathbb{Q}(\alpha)$ for some fixed complex root of unity α.

Let $\mathcal{N}$ denote a normal basis of $\mathcal{F}/\mathcal{K}$ defined as

$$\mathcal{N} = \{\alpha_0, \alpha_1, \ldots, \alpha_{r-1}\},$$

where $\alpha_i = \sigma_1^i(\alpha_0)$ for $i = 0, \ldots, r-1$.

As with any modular interpolation strategy, the method requires a bound s on the number of evaluation points. A suitable bound is given by the total degree of the expected resultant plus one, which is the sum of the degrees of the input

factors plus one, mirroring the commutative case. This is justified by the fact that, in Ore algebras, $\deg(fg) = \deg(f) + \deg(g)$ for any two skew polynomials f and g. If the base field does not contain enough distinct evaluation values (i.e. elements from different conjugacy classes), it may be necessary to extend the field appropriately. In practice, however, selecting a sufficiently large root of unity typically suffices to generate enough distinct values for interpolation.

Recall the identity for the resultant of two bivariate skew polynomials f and g in $\mathcal{F}[\theta_1;\sigma_1][\theta_2;\sigma_2]$ of degrees m and k, respectively:

$$\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(\mathrm{Res}_{\theta_2}(f,g)) = \mathrm{Res}_{\theta_2}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(f), \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(g)). \tag{5.5}$$

Assume the resultant R has the following generic form:

$$R = \sum_{i=0}^{s-1} c_i \theta_1^i, \tag{5.6}$$

where the coefficients $c_i \in \mathcal{F}$ are to be determined.

Fix a normal element $\alpha_0 \in \mathcal{F}$, and evaluate the right-hand side of Equation (5.5) at the points α_0^j for $j = 0, 1, \ldots, s-1$:

$$\begin{aligned}\mathrm{Res}_{\theta_2}(\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(f), \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(g))(\alpha_0^j) &= \left(\prod_{i=1}^{k} d_i(\sigma_1)\right)(\alpha_0^j) \\ &= (d_1(\sigma_1)d_2(\sigma_1)\cdots d_k(\sigma_1))(\alpha_0^j) \\ &= d_1^*(d_2^*(\cdots d_k^*(\alpha_0^j))),\end{aligned} \tag{5.7}$$

where $d_i^* = d_i(\sigma_1)$ denotes the operator evaluation at each diagonal entry in the triangular form of the σ-Sylvester matrix. The resulting scalar is denoted by $R^*(\alpha_0^j)$.

On the other hand, evaluate the generic form (5.6) of the resultant at the

same points α_0^j using operator evaluation:

$$\begin{aligned}\operatorname{eval}_{(\theta_1\text{-}\sigma_1)(\alpha_0^j)}(R) = R^*(\alpha_0^j) &= \sum_{i=0}^{s-1} c_i \sigma_1^i(\alpha_0^j) \\ &= \sum_{i=0}^{s-1} c_i \alpha_i^j, \quad \text{since } \alpha_i = \sigma_1^i(\alpha_0). \end{aligned} \tag{5.8}$$

Equating the two expressions (5.7) and (5.8) yields a linear system in the unknowns c_i. The system can be expressed as follows:

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_0 & \alpha_1 & \cdots & \alpha_{s-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_0^{s-1} & \alpha_1^{s-1} & \cdots & \alpha_{s-1}^{s-1} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{s-1} \end{pmatrix} = \begin{pmatrix} R^*(\alpha_0^0) \\ R^*(\alpha_0^1) \\ \vdots \\ R^*(\alpha_0^{s-1}) \end{pmatrix} \tag{5.9}$$

Solving this linear system recovers the coefficients $c_0, \ldots, c_{s-1}$, thereby yielding the explicit expression for the resultant R in the form (5.6).

5.1.5 The Evaluation and Interpolation Algorithm for Resultant Computation

The evaluation–interpolation method integrates several key components. The algorithm for computing the resultant

$$R(\theta_1) = \operatorname{Res}_{\theta_2}(f(\theta_1, \theta_2), g(\theta_1, \theta_2))$$

proceeds as follows:

Algorithm 4: Computation of the Resultant of Bivariate Skew Polynomials via Operator Evaluation–Interpolation

Input : Two bivariate skew polynomials

$$f(\theta_1,\theta_2),\ g(\theta_1,\theta_2) \in \mathcal{F}[\theta_1;\sigma_1][\theta_2;\sigma_2];$$

An integer bound $D \geq \deg_{\theta_1}\big(\mathrm{Res}_{\theta_2}(f,g)\big)$.

Output: The resultant $R(\theta_1) = \mathrm{Res}_{\theta_2}(f,g) \in \mathcal{F}[\theta_1;\sigma_1]$.

`/* Step 1: Select evaluation points (roots of unity) */`

$N := D+1$

Choose a prime $p > N$ and select a primitive p-th root of unity $\alpha = e^{2\pi i/p}$

Define the evaluation set

$\Delta = \{\alpha_j\}_{j=0}^{N-1}$ with $\alpha_j := \sigma_1^j(\alpha)$, for example $\sigma_1^j(\alpha) = \alpha^{2^j}$

`/* Step 2: Construct and evaluate the operator resultant */`

`/* 2.1: Operator evaluation */`

Apply the Ore morphism $\mathrm{eval}_{(\theta_1\text{-}\sigma_1)}$ to f and g:

$\widehat{f} := \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(f), \quad \widehat{g} := \mathrm{eval}_{(\theta_1\text{-}\sigma_1)}(g) \in A_1[\theta_2;\sigma_2]$,

where $A_1 = \mathcal{F}[\sigma_1;\circ]$

Construct the σ-Sylvester matrix

$S := \mathrm{Sylv}_{\theta_2}(\widehat{f},\widehat{g}) \in M_{m+n}(A_1)$

Triangularise S using unimodular row operations. Let

$d_1,\ldots,d_k \in A_1$ be the diagonal elements

Define the *operator resultant*

$R^* := d_1 \star d_2 \star \cdots \star d_k \in A_1$,

where "$\star$" denotes the ordered non-commutative (skew) product

`/* 2.2: Field evaluation at roots of unity */`

for $j := 0$ **to** $N-1$ **do**

  $R_j^* := d_1^*\big(d_2^*(\cdots d_k^*(\alpha_j)\cdots)\big)$

  `// This is nested operator evaluation`

end

`/* Step 3: Skew-polynomial interpolation */`

Assume the unknown resultant is of the form

$R(\theta_1) = \sum_{k=0}^{D} c_k \theta_1^k$

Form the interpolation system

$\sum_{k=0}^{D} c_k \sigma_1^k(\alpha_j) = R_j^*, \qquad j = 0,\ldots,N-1$

The associated coefficient matrix is the skew–Vandermonde $\big[\sigma_1^k(\alpha_j)\big]_{j,k}$

Solve for the coefficient vector

$(c_0,\ldots,c_D) \in \mathcal{F}(\alpha)^N$

`/* Step 4: Return result */`

return $R(\theta_1)$

5.1.6 Complexity Reduction via Modular Evaluation and Interpolation

The primary motivation for adopting the modular evaluation–interpolation framework is the mitigation of *intermediate expression swell*. In direct skew resultant computation, this phenomenon arises as a rapid growth in both coefficient bit-lengths and polynomial degrees during the symbolic row operations on the Sylvester matrix, rendering direct methods computationally infeasible for large inputs.

1. Performing computations directly over infinite domains such as $\mathbb{Q}$ involves arbitrary-precision arithmetic, where the cost of operations grows super-linearly with the bit-length of the operands. By reducing coefficients modulo a prime p (or a sequence of primes) and working over a finite field $\mathbb{F}_p$, the arithmetic cost becomes effectively constant ($O(1)$ machine instructions) relative to the coefficient size. The final result over the global ring is then recovered via the Chinese Remainder Theorem (CRT), thus avoiding the handling of massive intermediate integers.

2. Symbolic computation of the Dieudonné determinant within the skew polynomial ring $\mathbb{F}[\theta_1; \sigma_1]$ requires every elementary row operation to process non-commutative polynomials. This not only expands the degrees in θ_1 but also invokes complex skew-multiplication rules at every step. The evaluation–interpolation strategy avoids this by mapping the problem to the scalar domain:

    - *Evaluation Phase:* The matrix entries are evaluated at a set of carefully chosen points (specifically, operator-evaluations along σ_1-orbits). This transforms the symbolic matrix of skew polynomials into a set of scalar matrices over $\mathbb{F}_p$.
    - *Scalar Elimination:* The determinants of these scalar matrices are computed using standard Gaussian elimination. This step proceeds in

$O(N^3)$ time per evaluation point (where N is the matrix dimension) and, crucially, is entirely free of skew polynomial arithmetic.

- *Interpolation Phase:* The resultant polynomial is reconstructed from these scalar images using skew interpolation algorithms. The cost of this phase is polynomial in the number of evaluation points N_{eval}, typically scaling as $O(N_{\text{eval}}^2)$ for Newton-like interpolation.

By isolating coefficient growth (managed via modular arithmetic) from degree growth (managed via evaluation–interpolation), the total bit-complexity of the algorithm becomes polynomial in the input size. Furthermore, the reduction of the elimination step to a collection of independent scalar linear algebra problems exposes a natural task-level parallelism, which is exploited in the implementation described in subsequent chapters.

5.1.7 Examples

In this section, the resultant of two bivariate skew polynomials over the number field $\mathbb{Q}(\alpha)$ is computed using both the direct approach and the evaluation–interpolation method outlined in the preceding sections.

Example 5.1.9. *Let $\mathbb{Q}(\alpha)[\theta_1;\sigma_1][\theta_2;\sigma_2]$ denote a bivariate skew polynomial ring, where $\alpha = e^{2\pi i/p}$ with $p = 101$, and σ_1, σ_2 are $\mathbb{Q}$-automorphisms of $\mathbb{Q}(\alpha)$ defined by*

$$\sigma_1(\alpha) = \alpha^2, \qquad \sigma_2(\alpha) = \alpha^3.$$

The task is to compute the resultant of the following two bivariate skew polynomials with respect to θ_2:

$$\begin{cases} f = \alpha\,\theta_1\theta_2^2 + \alpha\,\theta_1 - 1, \\ g = \alpha^2\,\theta_1^2\theta_2 - \alpha. \end{cases} \tag{5.10}$$

Let $\mathrm{M} = \mathrm{Sylv}_{\theta_2}(f, g)$ be the σ-Sylvester matrix associated with f and g. In general form, this matrix is

$$\mathrm{Det}(M) = \mathrm{Det}(\mathrm{Sylv}_{\theta_2}(f,g)) = \begin{array}{c|ccc} f & a_2^{[0]} & a_1^{[0]} & a_0^{[0]} \\ \theta_2 g & b_1^{[1]} & b_0^{[1]} & \\ g & & b_1^{[0]} & b_0^{[0]} \end{array}$$

$$= \begin{vmatrix} a_2 & a_1 & a_0 \\ \sigma_2(b_1) & \sigma_2(b_0) & \\ & b_1 & b_0 \end{vmatrix},$$

which for the given example evaluates to

$$= \begin{vmatrix} \alpha\,\theta_1 & 0 & \alpha\,\theta_1 - 1 \\ \alpha^6\,\theta_1^2 & -\alpha^3 & \\ & \alpha^2\,\theta_1^2 & -\alpha \end{vmatrix}.$$

Applying row operations (see Lemma 3.6.9, Theorem 3.6.10, and Remark 3.6.11), the matrix is triangularised as follows:

$$\mathrm{Det}(M) = \begin{vmatrix} \alpha\,\theta_1 & 0 & \alpha\,\theta_1 - 1 \\ 0 & -\alpha^3 & -\alpha^6\theta_1^2 + \alpha^4\theta_1 \\ 0 & 0 & -\alpha^{14}\theta_1^4 + \alpha^6\theta_1^3 - \alpha \end{vmatrix}.$$

The direct computation of the resultant is thus

$$\begin{aligned}\mathrm{Det}(M) &= \prod_{i=1}^{3} d_i \\ &= \alpha\,\theta_1 \cdot (-\alpha^3) \cdot \left(-\alpha^{14}\theta_1^4 + \alpha^6\theta_1^3 - \alpha\right) \qquad (5.11)\\ &= \alpha\,\theta_1 \cdot \left(\alpha^{17}\theta_1^4 - \alpha^9\theta_1^3 + \alpha^4\right) \\ &= \alpha^{35}\theta_1^5 - \alpha^{19}\theta_1^4 + \alpha^9\theta_1. \qquad (5.12)\end{aligned}$$

Next, the same resultant is now computed using the evaluation-interpolation approach outlined in Theorem 5.1.6:

1. From the right-hand side of (5.5), the following composition is evaluated

$$d_1^*\left(d_2^*\left(d_3^*(w)\right)\right), \qquad (5.13)$$

for some p-th root of unity w. For instance, w can be taken as a power of α in the range $1 < \deg(w) < p$. First, compute

$$\begin{aligned}d_3^*(w) &= (-\alpha^{14}\sigma_1^4 + \alpha^6\sigma_1^3 - \alpha)(w) \\ &= -\alpha^{14}\sigma_1^4(w) + \alpha^6\sigma_1^3(w) - \alpha w \\ &= -\alpha^{14}w^{16} + \alpha^6 w^8 - \alpha w.\end{aligned}$$

Now apply this to (5.13):

$$\begin{aligned}d_1^*\left(d_2^*\left(d_3^*(w)\right)\right) &= \alpha \cdot \sigma_1\left(-\alpha^3(-\alpha^{14}w^{16} + \alpha^6 w^8 - \alpha w)\right) \\ &= \alpha \cdot \sigma_1\left(\alpha^{17}w^{16} - \alpha^9 w^8 + \alpha^4 w\right) \\ &= \alpha^{35}w^{32} - \alpha^{19}w^{16} + \alpha^9 w^2.\end{aligned}$$

This expression defines the *evaluated resultant polynomial*

$$R(w) = \alpha^{35}w^{32} - \alpha^{19}w^{16} + \alpha^9 w^2. \qquad (5.14)$$

2. Choose evaluation values from the normal basis

$$\mathcal{N} = \{\sigma_1^i(\alpha) \mid i = 0, \ldots, p-1\}.$$

3. To determine the number of evaluations required, use the degree bound: $\deg(d_1) + \deg(d_2) + \deg(d_3) + 1 = 6$. Hence evaluate (5.14) at

$$w_i = \sigma_1^i(\alpha), \quad i = 0, \ldots, 5$$

i.e. $w_0 = \alpha$, $w_1 = \alpha^2$, $w_2 = \alpha^4$, $w_3 = \alpha^8$, $w_4 = \alpha^{16}$, $w_5 = \alpha^{32}$, since $\sigma_1(\alpha) = \alpha^2$ and $\sigma_1(w_i) = w_{i+1}$.

4. Let V be a vector containing the evaluations $R(w_i)$:

$$R(w_0) = R(\alpha) = \alpha^{67} - \alpha^{35} + \alpha^{11}, \quad \text{etc.}$$

5. On the left-hand side of (5.5), assume the resultant has the form

$$R = c_5\theta_1^5 + c_4\theta_1^4 + c_3\theta_1^3 + c_2\theta_1^2 + c_1\theta_1 + c_0, \tag{5.15}$$

and determine the coefficients $c_0, \ldots, c_5$. From the interpolation condition $R^*(w_i) = \sum_{j=0}^{5} c_j \sigma_1^j(w_i)$, this yields the linear system:

$$Mx = V, \tag{5.16}$$

where $x = (c_0, \ldots, c_5)^T$ and M is the skew–Vandermonde matrix

$$M = \begin{pmatrix} \alpha & \alpha^2 & \alpha^4 & \alpha^8 & \alpha^{16} & \alpha^{32} \\ \alpha^2 & \alpha^4 & \alpha^8 & \alpha^{16} & \alpha^{32} & \alpha^{64} \\ \alpha^4 & \alpha^8 & \alpha^{16} & \alpha^{32} & \alpha^{64} & \alpha^{27} \\ \alpha^8 & \alpha^{16} & \alpha^{32} & \alpha^{64} & \alpha^{27} & \alpha^{54} \\ \alpha^{16} & \alpha^{32} & \alpha^{64} & \alpha^{27} & \alpha^{54} & \alpha^{7} \\ \alpha^{32} & \alpha^{64} & \alpha^{27} & \alpha^{54} & \alpha^{7} & \alpha^{14} \end{pmatrix}$$

Solving the system yields

$$\begin{pmatrix} 0 \\ \alpha^9 \\ 0 \\ 0 \\ -\alpha^{19} \\ \alpha^{35} \end{pmatrix},$$

which matches the direct result (5.12).

Example 5.1.10. *As an infinite-dimensional example, consider the problem of eliminating θ_1 from the algebra $\mathcal{F}[\theta_1; \sigma_1^*][\theta_2; \sigma_2^*]$, where $\mathcal{F} = \bigcup_{n\geq 1} \mathbb{Q}(\alpha_n)$, and each α_n is a primitive p^n-th root of unity for some fixed (odd) prime p.*

In this setting, $\mathcal{F}$ is the union of all p^n-th cyclotomic fields, each of which is a finite Galois extension. The full Galois group $G = \mathrm{Gal}(\mathcal{F}/\mathbb{Q})$ is a limit of finite Galois groups. An automorphism σ_n acts via

$$\sigma_n(\alpha_n) = \alpha_n^{a_n}, \quad \text{where } a_n \in \mathbb{Z} \text{ and } \gcd(a_n, p) = 1 \tag{5.17}$$

(see [39], Example 3.7), and satisfies compatibility:

$$\alpha_{n+1}^p = \alpha_n, \qquad \sigma_n|_{\mathbb{Q}(\alpha_{n-1})} = \sigma_{n-1}.$$

This permits the extension of each σ_n to a global automorphism $\sigma^* \in G$. For

example, setting $a_n = 2$ for all n, one obtains

$$\sigma^*(\alpha_n) = \alpha_n^2.$$

If m_n is the order of σ_n (i.e. $\sigma_n^{m_n} = \mathrm{id}$) , then

$$\sigma_n^{m_n}(\alpha_n) = \alpha_n^{2^{m_n}} = \mathrm{id}(\alpha_n) = \alpha_n, \quad \text{so } 2^{m_n} \equiv 1 \pmod{p^n}.$$

Hence $m_n \to \infty$ as $n \to \infty$, so σ^* is of infinite order.

Hachenberger [68] provides constructive techniques for building normal elements in such extensions (including those that are *completely normal*, meaning they remain normal over every intermediate subfield). The interpolation procedure of Example 5.1.9 can be adapted to this setting by working within a sufficiently large finite subfield $\mathcal{F}' \subset \mathcal{F}$ and repeating the same steps.

Chapter Summary

The preceding chapters have established a comprehensive framework for the theory and computation of resultants for bivariate skew polynomial systems. This study addressed the challenges inherent in non-commutative algebra by providing both a direct, theoretically sound definition of the resultant and a computationally efficient algorithm for its calculation.

Initially, a direct method for defining the bivariate skew resultant was presented, founded upon the Dieudonné determinant of the σ-Sylvester matrix derived from the polynomials. This approach formalised the skew resultant as an object in the Ore polynomial ring of the remaining indeterminate $\mathcal{F}[\theta_1; \sigma_1]$, and established its fundamental algebraic properties. These include its uniqueness modulo commutators, the well-definedness of its degree, and its crucial role as an elimination operator that annihilates common solutions of the original system. While this direct method is fundamental for theoretical understanding, its practical application can be limited by its computational intensity and the potential for intermediate expression swell, especially for polynomials of higher degrees.

To overcome these computational challenges, a second, more efficient modular strategy based on an evaluation-interpolation framework was developed. This alternative methodology reduces a complex bivariate problem into multiple, simpler univariate instances by evaluating one indeterminate at a series of carefully chosen points. After computing the resultants of these simpler univariate polynomials over the base field $\mathcal{F}$, a final interpolation step reconstructs the desired bivariate resultant. This involved addressing the specific challenges of evaluation in non-commutative settings (e.g. by selecting points from distinct conjugacy classes) and developing an effective skew polynomial interpolation scheme.

Together, these two approaches provide an adaptable and effective framework, offering both a direct algebraic construction for theoretical work and a practical, efficient algorithm for computational applications involving skew polynomial resultants.

This framework contributes to non-commutative symbolic computation in general and to elimination theory in particular. Based on resultant computations, the subsequent chapters explore practical applications of these properties, demonstrating the power of the resultant in designing novel and secure cryptographic schemes.

Chapter 6

Experimental Analysis of Bivariate Skew Polynomial Resultant Algorithms

This chapter presents an experimental evaluation of the bivariate skew polynomial resultant algorithms developed and discussed in the preceding chapters. Specifically, it focuses on the implementation and performance characteristics of two distinct approaches:

1. A *Direct Method*, based on the symbolic construction of a Sylvester-style matrix and its subsequent triangularisation via unimodular row operations as detailed in Algorithm 3.

2. An *Evaluation-Interpolation (Eval-Interp) Method*, which employs operator evaluation at roots of unity, as detailed in Algorithm 4.

The aim is to assess their practical performance and compare their computational efficiencies across various input parameters, such as polynomial degrees and coefficient sizes. The analysis highlights the impact of symbolic expression inherent in the Direct Method and contrasts it with the non-commutative evaluation framework which forms the basis of the Eval-Interp method.

6.1 Implementation and Experimental Framework

All experiments were carried out within the Maple symbolic computation environment (Version 2024) [97]. The implementation uses Maple's `Ore_algebra` package for defining the necessary algebraic structures and performing fundamental operations on skew polynomials.

6.1.1 Experimental Setup

This subsection summarises the algebraic environment, the method for generating random instances, and the specific evaluation strategy employed to experimentally compare the direct and evaluation–interpolation algorithms for skew polynomial resultants.

- **Algebraic Environment.** All experiments are conducted within the iterated Ore algebra

$$\mathcal{S} \;=\; \mathcal{R}_1[\theta_2;\sigma_2,\delta_2], \qquad \mathcal{R}_1 \;=\; \mathcal{F}[\theta_1;\sigma_1,\delta_1],$$

  where $\mathcal{F}$ serves as the base (skew) field. In the experimental configuration, $\mathcal{F}$ is the cyclotomic field extension $\mathcal{F} = \mathbb{Q}(\alpha)$, with α being a primitive 101^{st} root of unity ($\alpha = e^{2\pi i/101}$). The derivations are specialised to zero ($\delta_1 = \delta_2 = 0$), while the automorphisms are defined by their action on the root of unity:

$$\sigma_1(\alpha) = \alpha^2, \qquad \sigma_2(\alpha) = \alpha^3,$$

  and they act as the identity on $\mathbb{Q}$. Consequently, the resultants are reconstructed within the univariate skew polynomial ring $\mathcal{R}_1 = \mathbb{Q}(\alpha)[\theta_1;\sigma_1]$.

- **Polynomial Representation.** Bivariate skew polynomials $f(\theta_1,\theta_2)$ and $g(\theta_1,\theta_2)$ are represented using Maple's native expression format and are

subsequently handled as skew polynomial objects within the `Ore_algebra` package for non-commutative computations. This process ensures that all arithmetic operations, degree calculations, and Euclidean steps (where applicable) in full accordance with the non-commutative Ore multiplication rules.

- **Random Instance Generation.** Benchmark instances consist of pairs of random bivariate skew polynomials (f, g) constructed with specific partial degree constraints. For each test case:
  - The partial bi-degrees $(\deg_{\theta_1}(f), \deg_{\theta_2}(f))$ and $(\deg_{\theta_1}(g), \deg_{\theta_2}(g))$ are selected such that each component degree is less than or equal to 5.
  - The coefficients of non-constant terms are generated in the form $c \cdot \alpha^k$, where c is a random integer chosen from $[-100, 100]$ and the exponent k is a random integer from $[1, \ldots, 20]$. The main constant term for each polynomial is chosen independently from the integer interval $[-100, 100]$.

  This generation process ensures the creation of moderately sized but algebraically non-trivial skew systems, suitable for verifying correctness and measuring computational timing.

- **Evaluation Strategy.** For the evaluation–interpolation algorithm, the selection of evaluation points for the indeterminate θ_1 is determined by the degree bound of the resultant and the requirements for non-commutative interpolation:
  - The total number of evaluation points is set to

$$N_{\text{eval}} \;=\; D_{\theta_1} + 1, \qquad D_{\theta_1} = \deg_{\theta_1}(f)\deg_{\theta_2}(g) + \deg_{\theta_2}(f)\deg_{\theta_1}(g),$$

    which represents the upper bound for the degree of the resultant $\mathrm{Res}_{\theta_2}(f, g)$.

- The evaluation points are chosen as powers of the primitive root, specifically $\alpha_j = \alpha^{2^j}$ for $j = 0, \ldots, N_{\text{eval}} - 1$. Since $\sigma_1(\alpha) = \alpha^2$, this structured choice guarantees that the points $\{\alpha_j\}$ belong to distinct σ_1-conjugacy classes. This property is essential for enabling successful non-commutative interpolation.
- The implementation actively monitors for *unlucky* evaluation points (those that cause a reduction in the θ_2-degree of the evaluated polynomials) which could corrupt the computation of partial resultants. If such a point α_j is detected, it is discarded and replaced by the next candidate in the sequence, ensuring that only degree-preserving evaluations are used in the interpolation stage.

Figure 6.1 provides a schematic overview of the experimental workflow utilised for computing the skew resultant. This pipeline outlines the procedural stages, ranging from the initial algebraic setup and random instance generation to the configuration of evaluation points, algorithm execution, and final cross-verification of results.

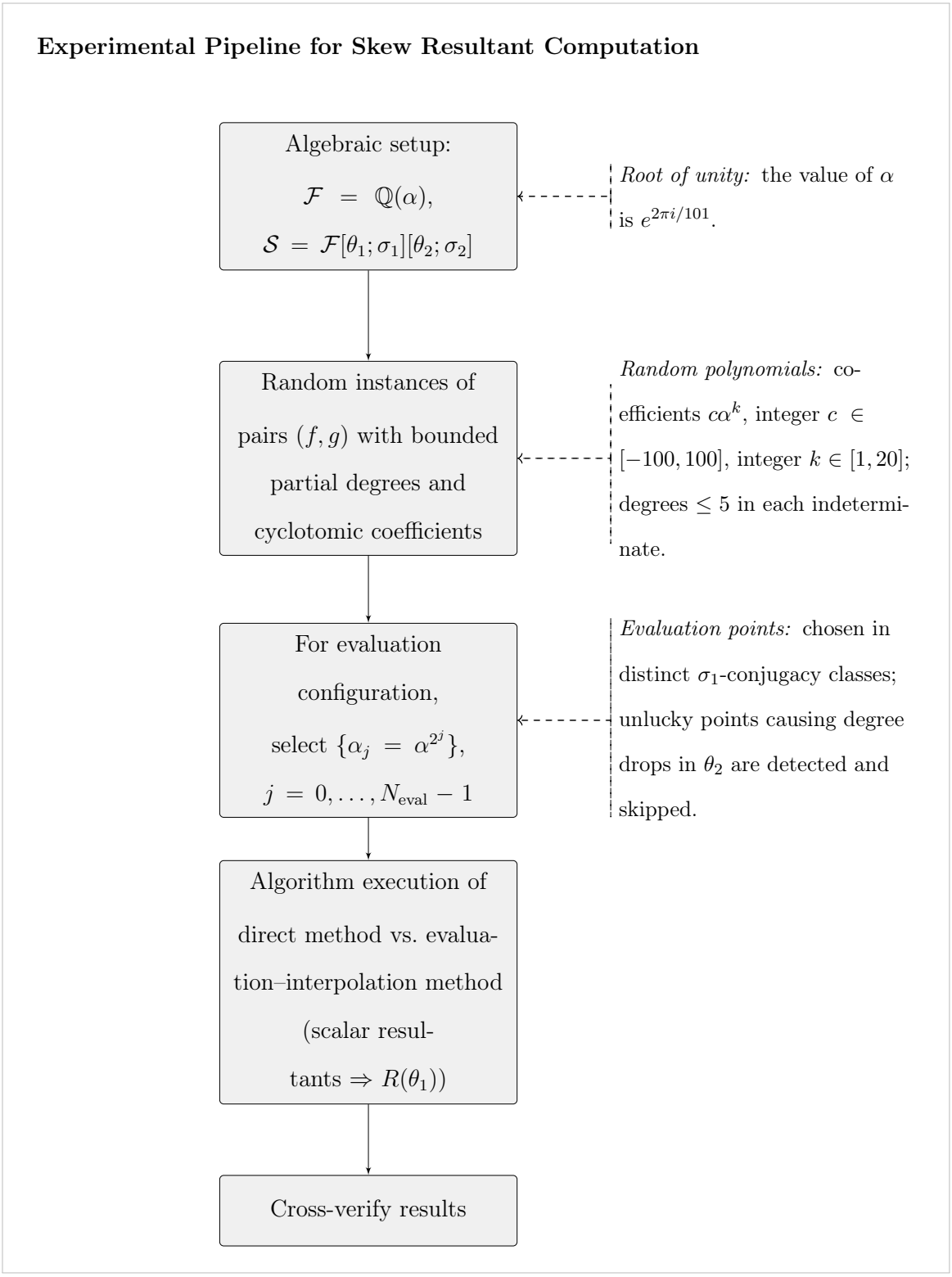

Figure 6.1: Experimental pathway for generating skew polynomial instances and applying the direct and evaluation–interpolation resultant algorithms.

6.1.2 Benchmarking Environment

All computational experiments were performed using Maple 2024 on a Linux machine with a multi-core processor system (Intel Core i7-8th generation, 6 cores @ 2.20 GHz), equipped with 12 GB of RAM, running Ubuntu 22.04. Execution times were measured in seconds using Maple's `time()` command and averaged over several consistent runs. Maple's `gc()` was invoked before each timed block to minimise interference from automatic memory management.

6.2 Benchmark Results and Analysis

This section presents and analyses the experimental performance results for the Direct and Eval-Interp methods.

6.2.1 Performance Data and Scalability

The timings are summarised in Table 6.1 and visualised in Figure 6.2. In addition to the raw CPU times for the Direct and Eval-Interp methods across degrees 1 to 5, the benchmark includes the following three derived metrics:

1. **Runtime Scaling:** This measures the relative speed, defined as the ratio of the execution time of the Eval-Interp method to that of the Direct method ($t_{\text{Eval}}/t_{\text{Direct}}$) which illustrates the growing computational advantage of the evaluation-interpolation strategy as the complexity of the input polynomials increases.

2. **Memory Usage (Direct vs. Eval-Interp):** These columns report the cumulative memory allocated (in Megabytes) by each method individually. Separating these metrics highlights the space complexity differences; specifically, it exposes the massive memory consumption of the Direct method due to expression swell, contrasted with the modest memory usage of the Eval-Interp method.

3. **Degree Scaling:** This measures the ratio of the degree of the computed resultant (in θ_1) to the sum of the partial degrees of the inputs (in θ_2), illustrating the inherent algebraic growth of the problem.

The experimental data reveals several key performance characteristics:

- For inputs with very small partial degrees, the Direct method is computationally competitive due to the setup overhead of the Eval-Interp method.
- As input degrees increase, the Direct method's performance degrades rapidly. This is primarily caused by *intermediate expression swell*, where symbolic manipulation of polynomial entries during triangularisation leads to coefficients and degrees far larger than the inputs. The "–" entries in Table 6.1 denote instances where the computation failed to complete within a reasonable timeframe, either by exceeding a predefined timeout (set to 1800 seconds in this study) or by exhausting available memory resources.
- The Eval-Interp method demonstrates significantly better and more predictable scalability. Its complexity is a well-behaved function of the number of evaluation points (N_{eval}) and the cost of the univariate resultants and final linear system solution.
- **Scalability and Complexity Classes:** The Eval-Interp method demonstrates significantly better scalability, operating within *polynomial time complexity*. Specifically, the cost is driven by solving the $N_{eval} \times N_{eval}$ interpolation system (typically $O(N_{eval}^3)$) and performing N_{eval} univariate resultant computations. Conversely, the Direct method is subject to *exponential intermediate expression swell*, where the size of coefficients during row reduction grows uncontrollably. This fundamental difference explains the *computational bottleneck* encountered by the Direct method at relatively low degrees (with practical failure already visible at $d = 4$), while the Eval-Interp method continues to scale predictably.

Table 6.1: Benchmark Timings, Memory Usage, and Scaling Metrics for Bivariate Skew Resultant Computation over $\mathbb{Q}(\alpha)$.

Degree		**Term**		**Resultant**		**Time (s)**		**Memory (MB)**		**Scaling**	
D_f	D_g	T_f	T_g	D_{θ_1}	N_{eval}	**Direct**	**Eval**	**Direct**	**Eval**	**Run**	**Deg**
1	1	2	2	1	2	0.001	0.005	0.14	0.59	5.000	0.500
1	2	2	2	3	4	0.002	0.004	0.27	0.44	2.000	1.000
2	1	2	2	2	3	0.002	0.002	0.24	0.24	1.000	0.667
2	2	3	2	4	5	0.003	0.009	0.38	1.08	3.000	1.000
3	1	3	2	4	5	0.119	0.007	12.42	1.05	0.059	1.000
3	2	3	3	5	6	2.728	0.040	469.28	7.42	0.015	1.000
4	1	3	2	6	7	98.779	0.297	19 162.71	28.20	0.003	1.200
4	2	3	2	7	8	337.136	25.875	116 553.88	239.76	0.077	1.167
5	3	3	3	6	7	–	13.063	–	3213.79	–	0.750
5	5	3	4	6	7	1936.15	193.084	693 818.02	166 218.42	0.100	0.600

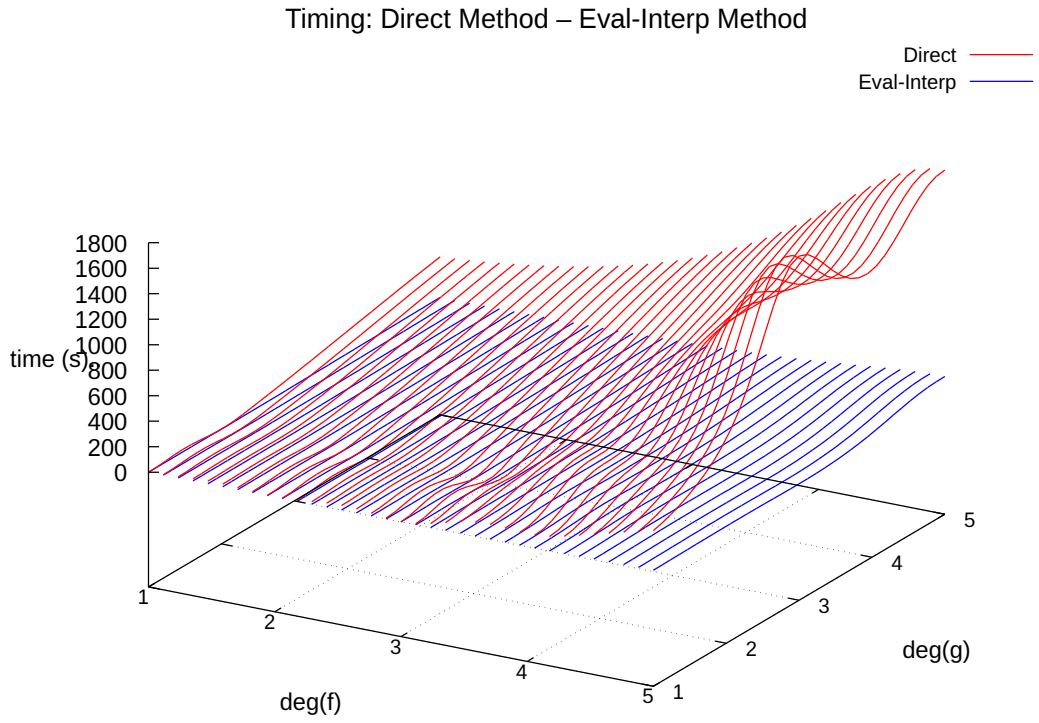

Figure 6.2: Performance benchmark comparing the Direct Method (red surface) and the Evaluation-Interpolation method (blue surface). The horizontal axes represent the degrees of the input polynomials, and the vertical axis is computation time in seconds. Note the non-monotonic "hill" on the red surface for the Direct method, indicating its performance sensitivity to specific algebraic structures, while the blue surface shows smoother scalability. Timings for the Direct method are capped at 1800 s (30-minute time-out).

6.2.2 Analysis of Non-Monotonic Performance

The non-monotonic "hill" observed in Figure 6.2 for the Direct Method is a genuine feature of its symbolic nature. The performance of such algorithms is highly path-dependent and sensitive to the specific algebraic relationships between intermediate expressions, not solely a function of initial input size.

- **The "Unlucky" Case (Peak of the Hill):** At certain degree combinations, the random generation can produce polynomials whose coefficients in the Sylvester matrix are algebraically complex. This can lead to transformation coefficients of high degree and large size during row reduction, causing dramatic intermediate expression swell and creating a performance

bottleneck.

- **The "Lucky" Case (Dip Following the Hill):** Conversely, a slightly different degree combination may coincidentally produce a "lucky" algebraic structure, such as a simple pivot or factors that allow for significant symbolic cancellation. This can dramatically reduce the computational cost, causing the observed dip in the performance surface.

In summary, the irregularity of the red surface reflects the structural sensitivity of direct symbolic elimination. The Eval-Interp method avoids this by mapping the problem to a series of more numerically-grounded computations over the base field $\mathcal{F}$, whose complexity scales predictably, as shown by the smooth blue surface.

6.2.3 Non-Monotonicity in Symbolic Performance

Symbolic computation in computer algebra systems does not, in general, exhibit smooth or monotone scaling behaviour as the nominal input size (for example, total degree or number of terms) increases. This effect is particularly pronounced for direct resultant computation, where the cost is governed less by the initial degrees of f and g and more by the structure of the *intermediate* expressions created during elimination.

Firstly, algorithms based on symbolic row operations over Ore polynomial rings are highly sensitive to *intermediate expression swell*. Small changes in degree or in the random coefficients of the input polynomials can lead to very different patterns of fill–in in the Sylvester matrix, different lengths of Euclidean reduction chains, and quite different growth of coefficient sizes. Consequently, a slightly larger instance may admit more cancellation or sparser intermediate matrices and thus run faster than a nominally smaller one, while an *unlucky* instance of lower degree may generate dense, high–degree intermediate polynomials and become disproportionately expensive.

Secondly, the internal heuristics of the computer algebra system contribute to non-monotonic timing behaviour. Operations such as automatic simplification,

normalisation of coefficients, term ordering choices, garbage collection, and memory reallocation are triggered in a data-dependent way. As a result, two inputs with very similar theoretical complexity can follow quite different execution paths: one might fit comfortably in cache and avoid major memory management overheads, while another triggers additional passes of simplification or more aggressive garbage collection, producing visible *spikes* in the measured running time.

Taken together, these factors explain why the timing curves for the purely symbolic (direct) method in Figure 6.2 do not increase smoothly with the degree parameters. The cost is governed by the algebraic and implementation details of the intermediate objects rather than by the nominal input size alone, leading to the observed irregular, non-monotone performance profile.

6.2.4 Implications for Cryptographic Protocol Design

The experimental scalability analysis has direct implications for the use of skew resultants in cryptographic protocol design. First, the observation that the direct Sylvester–triangularisation method suffers from severe intermediate expression swell, and becomes impractical already at modest degrees, indicates that it is unsuitable as the core computational engine for protocols requiring large security parameters. In contrast, the evaluation–interpolation method exhibits much more predictable, polynomial-time growth in running time as the degrees and number of participants increase, making it a realistic candidate for deployment in practice.

From a parameter-selection perspective, these results suggest that cryptographic schemes based on skew resultants should be realised within the domain where the evaluation–interpolation approach remains efficient, thus allowing the use of higher-degree polynomials and larger underlying fields without causing computational overload. This supports the use of skew-resultant constructions for threshold secret sharing and key-exchange protocols in settings where conventional symbolic elimination would be infeasible.

Moreover, the clear separation between an *offline* heavy computation phase

(in which the dealer or system initialiser computes the skew resultant) and the *online* phase (in which participants perform relatively light algebraic checks and reconstructions) aligns well with practical deployment scenarios. Resource-rich servers can accommodate the computational load of the resultant, whilst end-users or embedded devices incur only modest overhead during share verification and reconstruction. Overall, the experimental evidence validates the view that skew resultants, when computed via an evaluation–interpolation framework, provide a scalable algebraic primitive that can underpin cryptographic protocols with realistically chosen security parameters.

6.2.5 Verification of Computations

Implementing two distinct computational approaches provides the significant advantage of mutual cross-verification. In all test cases where both algorithms ran to completion, the resultant polynomials obtained were checked against one another. Their equivalence serves as an effective confirmation of the correctness of both implementations. This cross-check acts as a powerful, self-contained validation mechanism, providing strong practical evidence for the results, which is further supported by the formal mathematical proofs for each method presented in earlier chapters.

Chapter Summary

The experimental analysis confirms the efficiency of the evaluation-interpolation method for computing resultants of bivariate skew polynomials. The results demonstrate a significant performance advantage for this technique as the degrees and coefficient sizes of the input polynomials increase, highlighting its superior scalability over purely symbolic direct methods.

The ability to verify the methods (by comparing the two different methods by each other and ensure they both returns the same answer when input polynomials

are the same) provides strong confidence to the correctness of the methods.

While challenges concerning the choice of evaluation points (unlucky points, distinct conjugacy classes for certain interpolation types) are acknowledged, techniques involving careful point selection, working in field extensions if necessary, or using linear system-based interpolation can effectively address these.

Overall, the experimental findings would highlight that the evaluation-interpolation framework (supported by a solid theoretical foundation including Theorem 5.1.6) represents a practical and more efficient pathway for resultant computation and thus for elimination in non-commutative Ore algebras. This is particularly relevant for tackling larger and more complex systems where direct symbolic methods become computationally prohibitive.

Chapter 7

Secret Sharing Schemes Utilising Polynomial Resultants

7.1 Introduction to Secret Sharing

Today's digital environment is characterized by extensive volume of data and a growing dependence on interconnected global networks. Technologies such as artificial intelligence (AI), quantum computing (both its potential and the threats it poses to current cryptography), cloud infrastructures, and secure multiparty computation (MPC) are rapidly evolving [27, 124]. This technological advancement has led to an exponential growth in the volume of digital information generated, processed, and stored globally. At the same time, the sophistication and frequency of security threats have also escalated, making data protection methodologies not merely beneficial but critically essential to safeguard confidentiality, integrity, and availability of sensitive information. In this dynamic environment, cryptographic techniques remain at the core of efforts to secure digital assets, necessitating continuous innovation and adaptation to address emerging challenges.

This chapter focuses on the cryptographic technique of *secret sharing*, a powerful approach for distributing sensitive information in a secure and reliable manner. Secret sharing, independently developed by Shamir [120] and Blakley [8], involves

a trusted distributor known as a *dealer* dividing a secret into multiple pieces, called *shares*. These shares are then distributed among a set of *participants*. The main principle is that only predefined, authorised subsets of participants (usually meeting a certain threshold number) can collaboratively reconstruct the original secret by pooling their shares. Conversely, any unauthorised subset of participants, or any group smaller than the threshold, should obtain no information about the secret from their collective shares. Shamir's foundational scheme encodes the secret as the constant term of a polynomial over a finite field, with each share being a point on the polynomial's graph. Reconstruction is achieved through polynomial interpolation. Blakley's scheme takes a geometric approach where the secret is represented as the point of intersection of multiple hyperplanes in a multi-dimensional space, and shares correspond to individual hyperplanes. These foundational contributions laid the groundwork for extensive research and development in the field of cryptography and secure distributed systems.

The effectiveness and flexibility of secret sharing schemes have led to their adoption in a wide range of cryptographic applications [3, 23]. These include:

- **Secure Data Storage and Management:** Sensitive data can be protected by distributing its shares across multiple storage servers, ensuring that the compromise of a limited number of servers does not reveal the data [133].

- **Key Management:** Secret sharing is fundamental to secure key management, particularly in distributed systems, by dividing private keys or master keys into shares. This is crucial for cryptographic systems where a single point of failure for key storage is unacceptable. Examples include managing keys in dynamic environments using techniques such as Churn-Robust Proactive Secret Sharing (CHURP), which employs bivariate polynomials [98].

- **Blockchain and Distributed Ledger Technologies:** Combining secret

sharing with blockchain can enhance data integrity verification frameworks in cloud storage [69] and secure key management for cryptocurrency wallets or smart contracts.

- **Secure Multiparty Computation (MPC):** Secret sharing is a foundational primitive for MPC, enabling multiple parties to jointly compute a function over their private inputs without revealing those inputs to each other. Cramer et al. [41] detail efficient constructions for verifiable secret sharing which are instrumental in MPC protocols. Applications range from privacy-preserving data analysis to secure auctions and peer-to-peer energy trading [94].

- **Electronic Voting Systems:** The confidentiality and integrity of votes in electronic voting systems can be greatly enhanced using secret sharing methods, for instance, by sharing the final result or individual votes [7].

- **Visual Cryptography:** Secret sharing techniques are employed for image encryption, where an image is divided into shares (often printed on transparencies) such that stacking a qualified number of shares reveals the secret image [125].

- **Digital Signatures:** Schemes can protect private signing keys by dividing them into shares, ensuring that a signature can only be generated by a threshold number of participants. This is applied, for example, to secure SM2 private keys [45].

A common form of secret sharing is the (t, n)-threshold scheme, where a secret is distributed among n participants such that any group of t or more participants can collaboratively reconstruct the secret, while any group of fewer than t participants cannot obtain any information about the secret. Such schemes are crucial for ensuring fault tolerance.

Recent advancements have also explored lattice-based threshold secret sharing schemes [26] as a promising avenue for post-quantum cryptography. These

schemes are built upon the hardness of lattice problems to provide security against attacks by quantum computers, enabling demanding applications that require long-term security assurances [107].

Furthermore, research has utilised bivariate polynomials in secret sharing contexts, usually to avoid the need for secure channels during setup or to enable pairwise key computations among participants [9]. For example, Harn et al. [72] employed asymmetric bivariate polynomials for secure communication, while Hsu et al. [76] and Liu et al. [96] used symmetric bivariate polynomials, building upon Shamir's polynomial-based approach.

The reconstruction of the secret in most schemes relies on Lagrange's or Newton's interpolation methods after share distribution [54, 119]. However, alternative reconstruction techniques, such as those based on the Chinese Remainder Theorem (CRT) [81] or Euler's Theorem [25], have also been explored.

Despite the extensive volume of research and widespread application, security vulnerabilities and limitations persist in a number of existing secret sharing schemes. A major concern is the effectiveness against adversarial behaviour, particularly when adversaries can actively manipulate participants, submit false shares during reconstruction, or when the number of participants involved in reconstruction exceeds the minimum threshold t. For instance, some schemes based on bivariate polynomials have been shown to be vulnerable to attacks where the secret can be recovered with fewer than t shares or where adversaries can compromise the secret if more than t participants are involved in a reconstruction attempt [30, 70, 78]. Ding et al. [46] also demonstrated insecurities in other schemes where an adversary could reconstruct the secret using specific combinations of published information. Furthermore, recent work by Yang et al. [131] has employed resultant techniques to reduce the complexity of solving the systems of polynomial equations that arise from arithmetic-oriented (AO) cryptographic primitives, thus aiding potential attacks.

These findings highlights a critical and ongoing need for the exploration and

development of novel secret sharing schemes that offer enhanced security, particularly with built-in verification capabilities for share integrity and effective techniques for detecting and identifying adversarial actions. This chapter addresses this need by presenting a novel application of polynomial resultants (specifically, resultants of bivariate commutative polynomials in these initial constructions) for designing novel (t, n)-threshold secret sharing schemes. To the best of current knowledge this represents the first application of polynomial resultants to construct secret sharing schemes, thus establishing a new research direction.

While traditional approaches such as Shamir's scheme rely on a single polynomial for share generation, the schemes proposed herein utilises the algebraic properties of resultants derived from two bivariate polynomials. This *double-structured polynomial approach* offers a richer framework for encoding secrets and embedding security features, such as built-in verification and adversary detection.

The algebraic developments regarding resultants of skew (Ore) polynomials, detailed in earlier chapters (e.g. Chapter 4, 5) and in related works [113, 114], provide a potential pathway for extending these resultant-based cryptographic techniques to more complex, non-commutative algebraic settings in future research, possibly leading to schemes with even stronger security properties.

The main contributions of the research presented in this chapter are:

- The introduction and detailed development of two novel (t, n)-threshold secret sharing schemes based on polynomial resultants:
  - **Scheme 1 (Dealer-Side Scaling):** This scheme, presented in Section 7.5.2, focuses on participant efficiency by distributing numerical shares derived from resultant evaluations. It incorporates a verification process enabled by binding commitments that allows for early adversary screening.
  - **Scheme 2 (Participant-Side Scaling):** This scheme, detailed in Section 7.5.5, employs masked symbolic coefficient shares. This approach provides a double-structured polynomial framework aimed at

enhancing security assurance and offers potential advantages for post-quantum cryptographic considerations due to its design accommodating symbolic verification and post-quantum secure primitives.

- The development of a crucial supporting algorithm (Algorithm 5), presented in Section 7.4. This algorithm is designed to assist the dealer in constructing suitable bivariate polynomials whose resultant possesses the exact degree required for a (t, n)-threshold and correctly encodes the secret typically via a public shift, using techniques such as homogenisation and designed substitutions.

- Comprehensive formal analysis of the proposed schemes and algorithm. This includes mathematical proofs (Proofs 7.4.3, 7.5.1, and 7.5.8) that establish the correctness of the polynomial construction algorithm and the properties of the secret sharing schemes, including their threshold behaviour, verifiability, and adversary identification capabilities.

- Illustrative examples (e.g. Examples 7.4.4, 7.5.2, and 7.5.9) demonstrating the practical operation of the polynomial construction algorithm and both secret sharing schemes. These examples include scenarios with adversarial participants to showcase the effectiveness of the built-in detection techniques.

- A comprehensive comparison of the two proposed schemes (Section 7.6.4), highlighting their respective operational characteristics, computational requirements, security features, advantages, and trade-offs, supported by descriptive analysis with visual aids such as Table 7.6.4 and flowcharts (Figure 7.1).

The structure of this chapter is as follows: Section 7.2 reviews essential mathematical preliminaries concerning polynomial resultants relevant to cryptographic applications and the fundamentals of threshold secret sharing. Section 7.3 ex-

amines the specific properties of resultants that are particularly suitable for constructing these schemes. Section 7.4 presents the algorithm for generating the input bivariate polynomials and discusses its formal correctness. Section 7.5 details the main resultant-based secret sharing concept and elaborates on the two proposed schemes, including their operational stages and security arguments. Section 7.6 provides a comparative analysis of these schemes. Finally, Section 7.6.4 concludes this cryptographic investigation, summarising its contributions and outlining promising directions for future research in resultant-based cryptography.

7.2 Preliminaries for Resultant-Based Secret Sharing

This section provides the necessary background for understanding the novel secret sharing schemes developed in this chapter. It begins by revisiting key properties of polynomial resultants that are particularly relevant to this cryptographic study, followed by an overview of fundamental concepts in (t, n)-threshold secret sharing.

7.2.1 Relevant Properties of Polynomial Resultants

The resultant of two polynomials is a classical algebraic tool, primarily used in elimination theory and algebraic geometry to determine if two polynomials share common roots or factors without explicitly computing them. As detailed in Chapter 4, the resultant can be defined for polynomials over various coefficient rings, including commutative fields and, with appropriate generalisations such as the Dieudonné determinant, for skew polynomials. For the initial construction of the secret sharing schemes in this chapter, the focus is on resultants of bivariate commutative polynomials over a field $\mathcal{F}$.

Let $\mathcal{F}$ be a field. Consider two bivariate polynomials $F(x, y), G(x, y) \in \mathcal{F}[x, y]$

with positive degrees in x:

$$F(x,y) = a_m(y)x^m + a_{m-1}(y)x^{m-1} + \cdots + a_0(y) \tag{7.1}$$

$$G(x,y) = b_k(y)x^k + b_{k-1}(y)x^{k-1} + \cdots + b_0(y) \tag{7.2}$$

where $a_i(y), b_j(y) \in \mathcal{F}[y]$ are polynomials in y, and the leading coefficients $a_m(y) \not\equiv 0$ and $b_k(y) \not\equiv 0$. The resultant of $F(x,y)$ and $G(x,y)$ with respect to x, denoted by $\mathrm{Res}_x(F,G)$, is a polynomial $R(y) \in \mathcal{F}[y]$. It is defined as the determinant of the matrix $\mathrm{Sylv}_x(F,G)$, an $(m+k)\times(m+k)$ matrix whose entries are the coefficients $a_i(y)$ and $b_j(y)$. The following properties of the resultant are fundamental to its application in the proposed secret sharing schemes:

1. **Vanishing Condition and Common Factors (e.g. [19, Section 5.3]):** The resultant $R(y) = \mathrm{Res}_x(F,G)$ vanishes identically (i.e. $R(y) \equiv 0$) if and only if $F(x,y)$ and $G(x,y)$ have a common factor in $\mathcal{F}[x,y]$ that has a positive degree in x. If $\alpha_0 \in \mathcal{F}$ is a specific value such that neither $a_m(\alpha_0)$ nor $b_k(\alpha_0)$ is zero (i.e. the degrees in x are preserved upon substituting $y = \alpha_0$), then $R(\alpha_0) = 0$ if and only if $F(x,\alpha_0)$ and $G(x,\alpha_0)$ (now univariate polynomials in x) have a common root in x over an algebraic closure of $\mathcal{F}$. This property links the resultant to the existence of common solutions.

2. **Evaluation (Specialisation) Property (e.g. [38, Theorem 4, p. 516]):** Let $\alpha_0 \in \mathcal{F}$ be a specific value, and let $R(y) = \mathrm{Res}_x(F(x,y), G(x,y))$. If the degrees of $F(x,\alpha_0)$ and $G(x,\alpha_0)$ with respect to x remain m and k respectively (i.e. $a_m(\alpha_0) \neq 0$ and $b_k(\alpha_0) \neq 0$), then the evaluation of the resultant at α_0 equals the resultant of the evaluated polynomials:

$$R(\alpha_0) = (\mathrm{Res}_x(F,G))(\alpha_0) = \mathrm{Res}_x(F(x,\alpha_0), G(x,\alpha_0)).$$

This property is crucial for generating shares by evaluating polynomials and

for verifying share consistency. It forms the basis of linking the abstract resultant polynomial (which may encode the secret) to shareable values.

3. **Scaling Property (e.g. [14, Remark 3, p. A.IV.77]):** The resultant is homogeneous with respect to the coefficients of each polynomial. Specifically, if $a, b \in \mathcal{F}$ are constants (or elements from $\mathcal{F}[y]$ that do not depend on x), then:

$$\mathrm{Res}_x(a \cdot F(x,y), b \cdot G(x,y)) = a^k b^m \,\mathrm{Res}_x(F(x,y), G(x,y)).$$

More generally, if $F'(x,y) = \lambda(y)F(x,y)$ and $G'(x,y) = \mu(y)G(x,y)$, then

$$\mathrm{Res}_x(F', G') = (\lambda(y))^k (\mu(y))^m \,\mathrm{Res}_x(F, G).$$

This property is crucial for constructing verification processes within the secret sharing schemes, allowing for checks based on known scaling factors.

4. **Expression as a Polynomial Combination (Bezout-type Identity)(e.g. [61, Chaper 6]):** There exist polynomials $A(x,y), B(x,y) \in \mathcal{F}[x,y]$ such that

$$A(x,y)F(x,y) + B(x,y)G(x,y) = \mathrm{Res}_x(F,G)(y).$$

Furthermore, the degrees of A and B with respect to x are bounded: $\deg_x(A) < k$ and $\deg_x(B) < m$. This identity implies that the resultant $R(y)$ belongs to the ideal generated by $F(x,y)$ and $G(x,y)$ in $\mathcal{F}(y)[x]$ (the ring of polynomials in x with coefficients being rational functions in y). This property highlights the resultant's role in capturing information about common solutions and is fundamental to its elimination-theoretic nature.

These properties are utilised in the design of the cryptographic schemes presented later in this chapter.

7.2.2 Fundamentals of Threshold Secret Sharing

Secret sharing includes a set of cryptographic techniques that allow a dealer to distribute a secret s among a group of n participants in such a way that only authorised subsets of these participants can reconstruct the secret. But any unauthorised subset, however, cannot obtain any information about s.

Definition 7.2.1 ((t, n)-Threshold Secret Sharing Scheme [8, 120])**.** *A (t, n)-threshold secret sharing scheme, where $1 \leq t \leq n$, is a method by which a dealer divides a secret s into n pieces $s_1, s_2, \ldots, s_n$, called shares. Each share s_i is distributed to participant P_i. The scheme must satisfy the following two properties:*

1. ***Correctness (Reconstructability):*** *Any group of t or more participants can jointly reconstruct the secret s from their shares.*

2. ***Security (Privacy or $t-1$ Privacy):*** *Any group of fewer than t participants (i.e. $t-1$ or less) cannot obtain any information about the secret s from their collective shares. More formally, the joint distribution of any $t-1$ shares is independent of the secret s.*

The parameter t is called the threshold of the scheme.

The two most well-known threshold secret sharing schemes are:

- **Shamir's Secret Sharing Scheme [120]:** This scheme is based on polynomial interpolation. To share a secret $s \in \mathcal{F}_q$ (a finite field of size $q > n$), the dealer chooses a random polynomial $P(z) = a_{t-1}z^{t-1} + \cdots + a_1 z + s$ of degree $t-1$ over $\mathcal{F}_q$, where the coefficients $a_1, \ldots, a_{t-1}$ are chosen uniformly at random from $\mathcal{F}_q$. The secret s is embedded as the constant term $P(0)$. Each participant P_i receives a share (z_i, y_i), where $z_i \in \mathcal{F}_q \setminus \{0\}$ are distinct public points, and $y_i = P(z_i)$. Any t participants can uniquely determine the polynomial $P(z)$ using Lagrange interpolation (or other interpolation methods) with their t points (z_j, y_j), and then recover the secret $s = P(0)$. Any $t-1$ participants, having only $t-1$ points, cannot uniquely determine

$P(z)$ because for any possible secret value s', there exists a unique polynomial of degree $t-1$ passing through their $t-1$ points and $(0, s')$. Thus, they cannot find any information about s.

- **Blakley's Secret Sharing Scheme [8]:** This scheme is based on affine geometry. The secret s is represented as a coordinate of a point in a t-dimensional affine space over a finite field $\mathcal{F}_q$. Each share corresponds to a $(t-1)$-dimensional hyperplane that contains this secret point. The intersection of any t distinct hyperplanes (shares) uniquely determines the secret point, and thus the secret s. Any $t-1$ hyperplanes intersect in a line (or higher-dimensional affine subspace if $t-1 <$ dimension of intersection), which still contains infinitely many points (if q is large enough), thus not revealing the specific secret point.

The schemes developed in this chapter build upon algebraic principles related to polynomial resultants, aiming to provide (t, n)-threshold capabilities along with *built-in techniques* for share verification and adversary detection, which are crucial enhancements over other secret sharing schemes.

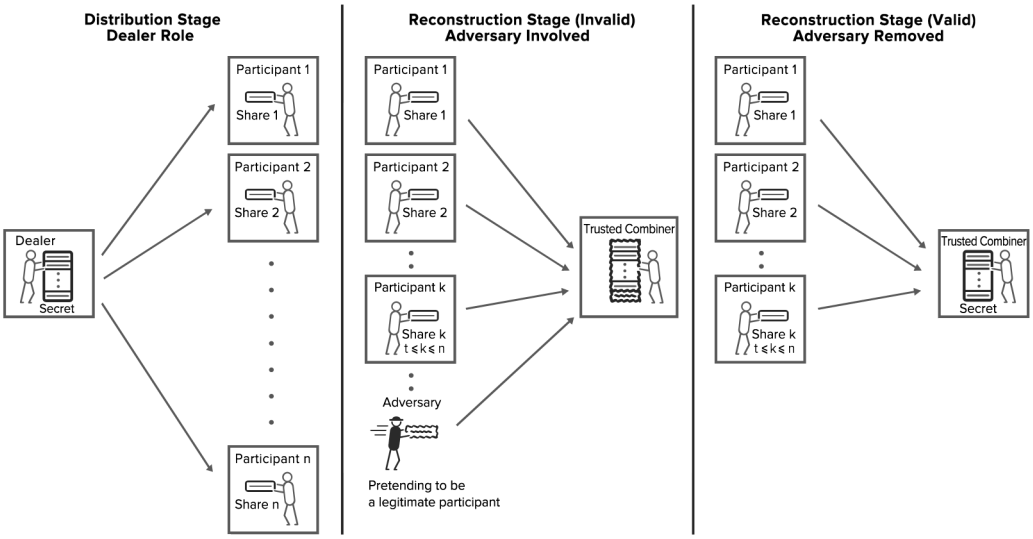

Figure 7.1: Illustration of an adversary submitting an invalid share, then being identified and removed, leading to a valid secret reconstruction independently from the dealer.

7.2.3 Polynomial Resultants for Secret Encoding

Polynomial resultants can function as a mechanism for the *algebraic encapsulation* of the secret. In this framework, two bivariate polynomials, denoted as $F(x, y)$ and $G(x, y)$, are constructed such that their resultant with respect to the variable x carries the secret in a controlled and verifiable manner.

The fundamental algebraic mechanism involves the elimination of x to yield a univariate polynomial in y, defined as $R(y) = \text{Res}_x(F(x, y), G(x, y))$. This resultant polynomial $R(y)$ encapsulates the precise algebraic conditions under which the system $F(x, y) = 0$ and $G(x, y) = 0$ possesses a common solution. Unlike standard Shamir schemes, where the secret is merely a coefficient of a single polynomial, the resultant establishes a *non-linear relationship* between the shares and the secret. This algebraic complexity significantly increases the difficulty for adversaries to manipulate shares without detection, as the unique structure of the resultant provides an inherent consistency check that is essential for verification.

The subsequent sections discuss this principle in detail; the next section first discuss the fundamental properties of the resultant required for this application, followed by the presentation of the polynomial construction algorithms and the proposed secret sharing schemes.

7.3 Key Resultant Properties for Secret Sharing Scheme Construction

The algebraic properties of polynomial resultants, as outlined in Section 7.2.1, are not merely theoretical constructs but offer practical tools that can be utilised for cryptographic purposes. This section elaborates on how specific properties are particularly well-suited to the design and operation of the novel secret sharing schemes proposed in this work.

7.3.1 The Evaluation (Specialisation) Property: Enabling Share Generation and Reconstruction

The evaluation property states that, under non-degenerate conditions, evaluating the resultant polynomial $R(y) = \text{Res}_x(F(x,y), G(x,y))$ at a point α_0 yields the same value as first evaluating $F(x,y)$ and $G(x,y)$ at α_0 to get $F(x,\alpha_0)$ and $G(x,\alpha_0)$, and then computing their resultant:

$$(\text{Res}_x(F,G))(\alpha_0) = \text{Res}_x(F(x,\alpha_0), G(x,\alpha_0)). \tag{7.3}$$

Applicability to Scheme Construction: This property is fundamental for both distributing and recovering the secret.

- **Share Generation:** If a secret s is encoded within the resultant polynomial $R(y)$ (specifically $s = R(\beta) + \delta_{\text{shift}}$), then shares for participants can be generated by evaluating $R(y)$ at distinct points $\alpha_i \in \mathcal{F}$. Each participant P_i would receive the pair $(\alpha_i, R(\alpha_i))$ as their share. Alternatively, as explored in Scheme 2 (Participant-Side Scaling), masked symbolic coefficient shares can be distributed. The evaluation property ensures that once these shares are unmasked to recover F and G, the locally computed resultant $R^*(y)$ is consistent with the dealer's original intent.

- **Secret Reconstruction:** If $R(y)$ is a polynomial of degree $t-1$ (a common choice for (t,n)-threshold schemes), then knowledge of t distinct pairs $(\alpha_i, R(\alpha_i))$ allows participants to uniquely reconstruct $R(y)$ using polynomial interpolation (e.g. Lagrange interpolation). Once $R(y)$ is reconstructed, the embedded secret s can be recovered by evaluating $R(\beta)$ and adding the public shift δ_{shift}.

The evaluation property thus provides a direct link between the algebraic object encoding the secret (the resultant polynomial) and the shareable, reconstructable pieces of information.

7.3.2 The Scaling Property: Facilitating Verification and Adversary Detection

The scaling property of resultants, $\text{Res}_x(aF, bG) = a^{d_2}b^{d_1}\,\text{Res}_x(F, G)$, is essential for embedding verification capabilities into the secret sharing schemes.

Applicability to Scheme Construction: This property allows for the creation of checkable relationships between shares themselves or between shares and publicly available commitments.

- **Share Verification and Consistency Checks:** A dealer can generate a primary pair of polynomials (F, G) whose resultant $R(y)$ encodes the secret. The dealer can then apply scalars a, b to generate a scaled resultant $R'(y) = rR(y)$, where $r = a^{d_2}b^{d_1}$. In the *Dealer-Side Scaling scheme* (Scheme 1), the dealer publishes binding commitments to the scaled evaluations $V_i' = R'(\alpha_i)$. A participant can verify their private share Sh_i by checking if the ratio $\rho_i = V_i'/Sh_i$ matches the expected scaling factor r. Alternatively, in the *Participant-Side Scaling scheme* (Scheme 2), participants use the scaling factor r as an algebraic filter to verify that their locally reconstructed resultant $R^*(y)$ satisfies $V_i' = rR^*(\alpha_i)$, ensuring structural consistency.

- **Adversary Identification:** If a participant submits a fraudulent share Sh_i^* during reconstruction, it will violate the expected scaling relationship. For instance, in Scheme 1, a tampered share will yield $\rho_i \neq r$. In Scheme 2, a corrupted share will result in a reconstructed resultant R^* that fails the algebraic filter check against the committed V_i'. A cross-checking process during reconstruction can pinpoint the source of inconsistency, thus identifying the dishonest participant(s).

The scaling property thus provides a powerful and algebraically natural way to build integrity checks and adversary detection directly into the structure of the secret sharing mechanism.

7.3.3 The Vanishing Condition and Polynomial Combination Property (Theoretical Foundation)

While perhaps less directly used in the operational steps of share generation/reconstruction, these properties provide essential theoretical support.

- **Vanishing Condition:** This property ensures that the resultant $R(y)$ is a meaningful algebraic object that truly reflects the presence or absence of a shared algebraic structure between $F(x, y)$ and $G(x, y)$. For secret sharing, it is crucial that $R(y)$ is not identically zero unless F and G share a common factor in x for all y. The construction algorithm for F and G (Algorithm 5) must ensure that $R(y)$ is non-trivial and correctly encodes the secret.

- **Polynomial Combination Property ($A \cdot F + B \cdot G = R$):** This highlights that the resultant $R(y)$ lies in the ideal generated by F and G. This structural property supports the consistency of the secret encoding; any information derived from F and G having common roots is algebraically tied to F and G themselves. This is relevant for arguments about the algebraic security of the scheme, ensuring that the secret encoded in $R(y)$ cannot be easily extracted or forged without breaking the underlying structure of F and G.

Together, these properties of polynomial resultants provide an effective and flexible algebraic framework, enabling the design of novel secret sharing schemes with sophisticated features beyond those offered by traditional single-polynomial approaches. The subsequent sections will detail how these properties are specifically utilised in the proposed schemes.

7.4 Construction of Bivariate Polynomials for Resultant-Based Secret Sharing

The efficiency and security of the proposed resultant-based secret sharing schemes depend critically on the careful construction of the input bivariate polynomials, $F(x, y)$ and $G(x, y)$, both elements of $\mathcal{F}[x, y]$ where $\mathcal{F}$ is a (finite) field. Their resultant with respect to x, $R(y) = \text{Res}_x(F, G)$, must possess specific structural properties to be suitable for encoding a secret s within a (t, n)-threshold framework. This section details the objectives for such a construction, discusses the conceptual algebraic techniques involved, and outlines a procedural approach for Algorithm 5 designed for this purpose.

7.4.1 Algorithm Objectives and Design Considerations

Algorithm 5 is intended to provide the dealer with a systematic method to generate $F(x, y)$ and $G(x, y)$ satisfying the following crucial properties:

1. **Correct Resultant Degree:** The resultant polynomial $R(y) = \text{Res}_x(F, G)$ must have an exact degree of $t - 1$. This degree determines that exactly t shares (evaluations of $R(y)$) are necessary and sufficient to uniquely reconstruct $R(y)$ using polynomial interpolation, aligning with the (t, n)-threshold requirement.

2. **Secure Secret Encoding:** The secret $s \in \mathcal{F}$ must be securely encoded within $R(y)$. Typically, this is achieved by setting the constant term of $R(y)$ to be a function of s, i.e. $R(\beta) = s - \delta_{\text{shift}}$, where δ_{shift} is a publicly known shift. This allows s to be recovered once $R(y)$ is reconstructed.

3. **Non-Triviality and Effectiveness:** The polynomials $F(x, y)$ and $G(x, y)$ should be constructed such that $R(y)$ is not identically zero (unless s and all other coefficients are zero, which is a trivial case). Furthermore, $F(x, y)$ and $G(x, y)$ should not share a common factor in a way that would compromise

the scheme. The construction should also ensure that $R(y)$ is well-behaved for evaluation.

4. **Sufficient Randomness/Unpredictability:** While not explicitly an output of the resultant's algebraic structure, the construction process should ideally allow for enough flexibility or randomness in choosing $F(x, y)$ and $G(x, y)$ so that they are not easily guessable or related in a trivial way, which could potentially be exploited by adversaries.

7.4.2 Conceptual Algorithmic Techniques

To achieve the above objectives, several algebraic techniques can be employed:

- **Degree Control via Polynomial Structure:** The degrees of $F(x, y)$ and $G(x, y)$ with respect to both x and y directly influence the degree of their resultant $R(y)$. Recall that $\deg_y(R(y)) \leq \deg_x(F) \cdot \deg_y(G) + \deg_x(G) \cdot \deg_y(F)$. The algorithm needs to choose these degrees carefully. For simplicity and controllability, one usually chooses low degrees for F and G in x, for example, $m = \deg_x(F) = 1$ and $k = \deg_x(G) = 1$.

- **Coefficient Assignment and System Solving:** One can define $F(x, y)$ and $G(x, y)$ with coefficients (which are themselves polynomials in y) that include some indeterminates. The Sylvester matrix is then formed symbolically, and its determinant $R(y)$ will have coefficients that are expressions in these indeterminates. By equating these expressions with the coefficients of a target resultant polynomial $R_{\text{target}}(y)$ (of degree $t-1$ and encoding s), one obtains a system of algebraic equations.

- **Constructive Approach for Fixed Degrees in x:** A more direct constructive method is often preferred for simplicity. For instance, if $m = 1$ and $k = 1$: Let $F(x, y) = a_1(y)x + a_0(y)$ and $G(x, y) = b_1(y)x + b_0(y)$. Their resultant is $\text{Res}_x(F, G) = a_1(y)b_0(y) - a_0(y)b_1(y)$. The dealer defines a target resultant $R_{\text{target}}(y)$ of degree $t - 1$ that embeds the se-

cret. The dealer then needs to find $a_1(y), a_0(y), b_1(y), b_0(y)$ such that $a_1(y)b_0(y) - a_0(y)b_1(y) = R_{\text{target}}(y)$. To ensure $\deg_y(R(y)) = t - 1$, the degrees of $a_i(y)$ and $b_j(y)$ must be appropriately chosen. For example:

- Choose $a_1(y)$ and $b_1(y)$ to be simple, non-zero constants, e.g. $a_1(y) = 1, b_1(y) = 1$.
- Then the condition becomes $b_0(y) - a_0(y) = R_{\text{target}}(y)$.
- The dealer can choose $a_0(y)$ as a random polynomial in y of degree $t - 1$. Then $b_0(y)$ is determined as $R_{\text{target}}(y) + a_0(y)$.

This construction provides $F(x, y)$ and $G(x, y)$ that yield the desired resultant. In Scheme 1, the values $R(\alpha_i)$ are distributed as shares. Alternatively, in Scheme 2, the coefficients of these constructed polynomials are distributed as *masked symbolic coefficient shares*, allowing participants to reconstruct F and G and subsequently compute $R(y)$.

- **Homogenisation (Conceptual):** While homogenisation is a known technique in algebraic geometry, its application in this context is primarily used as a tool for construction. Here, it refers to the process of balancing the degrees of the terms in F and G to ensure that their resultant $R(y)$ behaves predictably, particularly with respect to its final degree.

- **Designed Substitution for Constant Term Control:** If it is essential that $R(0) \neq 0$, and a preliminary construction yields $R(0) = 0$, a transformation $y \mapsto y + \gamma$ for some non-zero $\gamma \in \mathcal{F}$ can be applied. This technique corresponds to the previously mentioned public shift δ_{shift}.

7.4.3 Achieving Exact Resultant Degree

While bounds on the degree of a resultant polynomial are readily available in the literature (e.g. [112]), the cryptographic schemes developed in this study require the resultant to have an *exact degree*, not merely an upper bound. To achieve

this, specific conditions must be applied to the input polynomials, particularly when using homogenisation as a construction technique.

Standard references (e.g. Lemma 3.7 in [84]) generally state that the resultant of two homogeneous polynomials has a degree equal to the product of their total degrees. However, this holds only under an additional, crucial condition. Without this condition, the resultant degree can be smaller than expected. For instance, consider the polynomials $F_0(x, y) = x + y$ and $G_0(x, y) = xy + 1$. Their respective homogeneous forms (using z as the homogenising variable) of total degree 2 are:

$$F_0^{(h)}(x, y, z) = xz + yz$$
$$G_0^{(h)}(x, y, z) = xy + z^2$$

Although both $F_0^{(h)}$ and $G_0^{(h)}$ have a total degree of 2, their resultant with respect to x is $z^3 - y^2z$, which has a total degree of 3 in y, z, not $2 \times 2 = 4$.

As established in standard algebraic geometry literature (e.g. [40, pp. 454-455] or [99, Theorem 5.4, p. 52]), for the resultant's degree to be exactly the product of the total degrees, the leading coefficients with respect to the elimination variable must be non-zero constants after projective specialisation. This is captured by the following lemma.

Lemma 7.4.1. *Let $\mathcal{F}$ be a field, and let $F, G \in \mathcal{F}[x, y, z]$ be non-constant homogeneous polynomials of total degrees d_1 and d_2 respectively. Suppose that F and G share no non-constant common factor in $\mathcal{F}[x, y, z]$, and that their leading coefficients with respect to x are non-zero constants, ensured by the conditions:*

$$F(1, 0, 0) \neq 0 \quad \textit{and} \quad G(1, 0, 0) \neq 0.$$

Then $\mathrm{Res}_x(F, G)$ is a non-zero homogeneous polynomial in the variables y and z of total degree exactly $d_1 d_2$.

Remark 7.4.2. *The conditions $F(1, 0, 0) \neq 0$ and $G(1, 0, 0) \neq 0$ directly imply that the degree of each polynomial with respect to x is equal to its total degree (i.e.*

$\deg_x(F) = d_1$ *and* $\deg_x(G) = d_2$*). If* $F(1,0,0) = 0$ *for a homogeneous polynomial* F *of total degree* d_1*, it would mean that every term in* F *must contain a factor of* y *or* z*. But in that case, evaluating at* $(1,0,0)$ *would necessarily yield* 0*, creating a contradiction. Thus, for* $F(1,0,0)$ *to be non-zero, the term* $c \cdot x^{d_1}$ *(with* $c \neq 0$*) must be present in* F*. The same logic applies to* G*.*

This theoretical result is the foundation for Algorithm 5, which is designed to construct polynomials that satisfy these conditions, thus guaranteeing a resultant of an exact, predictable degree.

7.4.4 Polynomial Construction via Homogenisation

The generation of suitable bivariate polynomials is a critical prerequisite for the resultant-based secret sharing schemes. The process detailed in Algorithm 5 provides a deterministic method for this task, employing homogenisation followed by a specific dehomogenisation step.

The process begins (Step 1) by choosing two random polynomials $F_0(x, y)$ and $G_0(x, y)$ such that their total degree is equal to their degree in x, denoted by d_1 and d_2 respectively. In the homogenisation stage (Step 2), an auxiliary variable z is introduced to transform these into homogeneous polynomials $F_0^{(h)}(x, y, z)$ and $G_0^{(h)}(x, y, z)$. This is achieved by multiplying each term $c_{ij}y^i x^j$ by the appropriate power of z (e.g. in the form $z^{d_1-(i+j)}$ for F_0) to ensure that every term has the required total degree. For example, if $F_0(x, y) = x^3 + 2xy + 1$ (where $\deg(F_0) = \deg_x(F_0) = d_1 = 3$), its homogenised form is

$$F_0^{(h)}(x, y, z) = x^3 + 2xyz + z^3.$$

Dehomogenisation is then performed via the substitution $z = \alpha_1 y + \alpha_2$, where $\alpha_1, \alpha_2 \in \mathcal{F} \setminus \{0\}$. This linear substitution into a homogeneous polynomial is a crucial step; it ensures that the resulting non-homogeneous polynomial retains a structure that guarantees a resultant of the maximal possible degree and also

ensures a non-zero constant term. This latter property is particularly crucial for the cryptographic application, as the constant term is the designated location for encoding the secret s. For simplicity, the specific substitution $z = y + 1$ is adopted (Step 3) to derive the final polynomials $F(x, y)$ and $G(x, y)$.

This choice of dehomogenisation, combined with the requirement that the initial polynomials $F_0(x, 0)$ and $G_0(x, 0)$ are coprime (Step 1), is a design feature. The coprimality of the polynomials when $y = 0$ ensures that the resultant does not vanish at $y = 0$. The specific dehomogenisation $z = y+1$ translates this non-vanishing property from the homogenised space to the constant term of the final resultant polynomial. This guarantees that $R(y) = \mathrm{Res}_x(F, G)$ has a non-zero constant term (i.e. $R(0) \neq 0$), as established in the proof of Lemma 7.4.3.

Although the substitution $z = y+1$ could, in principle, be applied *after* computing the resultant of the homogenised polynomials $\mathrm{Res}_x(F_0^{(h)}, G_0^{(h)})$, applying it *beforehand* (Step 3) significantly simplifies the resultant computation (Step 4). This pre-substitution reduces the problem to computing the resultant of two polynomials in x whose coefficients are univariate polynomials in y. Consequently, the corresponding Sylvester matrix entries are polynomials in y only, rather than bivariate polynomials in y and z. This simplification is justified by the evaluation property of resultants, which ensures that specialising a variable before computing the resultant is equivalent to computing the resultant and then specialising the variable.

As established in the proof of Lemma 7.4.3, the resulting polynomial $R(y)$ possesses both a non-zero leading coefficient $c_{d_1 d_2}$ and a non-zero constant term $c_0 = R(0)$, ensuring its degree is exactly $d_1 d_2$. This exact degree control is essential for establishing the threshold of the resulting secret sharing scheme. Finally, the public shift δ_{shift} is computed as $\delta_{\text{shift}} = s - R(0)$ (Step 5) to ensure that $R(0) + \delta_{\text{shift}}$ equals the desired secret s. In the final step of Algorithm 5, the polynomials $F(x, y)$ and $G(x, y)$, their resultant $R(y)$, and the shift δ_{shift} are returned.

This procedure provides a deterministic method for generating suitable polynomials for resultant-based secret sharing schemes, thus eliminating the need for trial-and-error approaches during this setup phase.

7.4.5 Algorithm for Constructing Bivariate Polynomials and Its Correctness

Algorithm 5: Construct Bivariate Polynomials for Resultant-Based Secret Sharing

Input : A (finite) field $\mathcal{F}$, integers d_1 and d_2 (the degrees of F and G w.r.t. x), and $s \in \mathcal{F}$ (the desired secret).

Output: Polynomials $F, G \in \mathcal{F}[x, y]$ and shift $\delta_{\text{shift}} \in \mathcal{F}$ such that $\deg_x(F) = d_1$, $\deg_x(G) = d_2$, and the resultant $R(y) = \text{Res}_x(F, G)$ satisfies $\deg_y(R) = d_1 d_2$, $R(0) \neq 0$, and $R(0) + \delta_{\text{shift}} = s$.

`/* Step 1: Choose initial coprime polynomials */`

1 Choose random polynomials $F_0(x, y)$ and $G_0(x, y)$ over $\mathcal{F}$

`/* with` $\deg_x(F_0) = \deg(F_0) = d_1$ `and` $\deg_x(G_0) = \deg(G_0) = d_2$ `*/`

`/* where` $F_0(x, 0)$ `and` $G_0(x, 0)$ `are coprime (ensures` $R(0) \neq 0$`) */`

`/* Step 2: Homogenise to set degree */`

2 Homogenise F_0 and G_0 to obtain $F_0^{(h)}(x, y, z)$ and $G_0^{(h)}(x, y, z)$

`/* Step 3: Dehomogenise step */`

3 Substitute $z := y + 1$ into $F_0^{(h)}$ and $G_0^{(h)}$ to obtain $F(x, y)$ and $G(x, y)$

`/* Step 4: Compute the resultant */`

4 Compute $R(y) := \text{Res}_x\big(F, G\big)$

`/* The resultant will have the form` $c_{d_1 d_2} y^{d_1 d_2} + \cdots + c_1 y + c_0$ `*/`

`/* Step 5: Compute the shift */`

5 Compute $\delta_{\text{shift}} := s - c_0$

`/* Step 6: Return results */`

6 **return** $F(x, y), G(x, y), R(y), \delta_{\text{shift}}$

The following lemma formally proves the correctness of Algorithm 5 for con-

structing the required bivariate polynomials. An illustrative example of its operation follows.

Lemma 7.4.3 (Correctness of Algorithm 5)**.** *Let d_1 and d_2 be positive integers, and let $\mathcal{F}$ be a (finite) field. Suppose Algorithm 5 is performed over $\mathcal{F}$ with input (d_1, d_2, s), where $s \in \mathcal{F}$ is the desired secret. Then the polynomials*

$$F(x, y), \quad G(x, y), \quad R(y)$$

returned by the algorithm, together with the public shift δ_{shift}, satisfy the following conditions:

1. $\deg_x(F) = d_1$ *and* $\deg_x(G) = d_2$.
2. $\text{Res}_x\big(F, G\big)$ *is a univariate polynomial $R(y)$ of degree $d_1 d_2$ in y.*
3. *The constant term $R(0)$ is non-zero and satisfies $R(0) + \delta_{\text{shift}} = s$.*

Proof. Let $s \in \mathcal{F}$ be the desired secret. The proof demonstrates that the polynomials and shift returned by Algorithm 5 satisfy each of the three stated properties:

1) **Degree in x:** In Step 1 of the algorithm, initial polynomials

$$F_0(x, y), \quad G_0(x, y)$$

are chosen such that $\deg_x(F_0(x, y)) = d_1$ and $\deg_x(G_0(x, y)) = d_2$.

Next, $F_0(x, y)$ and $G_0(x, y)$ are homogenised (Step 2) by introducing an auxiliary variable z. This process modifies each term by multiplying with an appropriate power of z such that the resulting polynomials $F_0^{(h)}(x, y, z)$ and $G_0^{(h)}(x, y, z)$ become homogeneous (i.e. all terms within $F_0^{(h)}$ have the same total degree, and similarly for $G_0^{(h)}$). This homogenisation step is performed to utilise established properties of resultants of homogeneous polynomials, such as those described in Lemma 7.4.2, particularly concerning the degree and structure of the resultant.

Subsequently, in Step 3, dehomogenisation occurs by substituting $z = y + 1$. This designed substitution does not reduce the degree of either polynomial in x, because the terms of the highest degree in x (which originated from F_0 and G_0) do not involve z in such a way that they could cancel or vanish upon setting $z = y + 1$. Consequently, the returned polynomials

$$F(x, y), \quad G(x, y)$$

retain the required degrees in x, with $\deg_x(F) = d_1$ and $\deg_x(G) = d_2$.

2) **Degree of resultant:** Let $F_0^{(h)}(x, y, z)$ and $G_0^{(h)}(x, y, z)$ be the homogenised polynomials from Step 2 of the algorithm. It is known that $\mathrm{Res}_x\big(F_0^{(h)}, G_0^{(h)}\big)$ is a homogeneous polynomial of total degree $d_1 d_2$ in the variables y and z (by Lemma 7.4.1). Since $d_1, d_2 > 0$, $d_1 d_2 > 0$. The substitution $z = y + 1$ (Step 3) transforms this homogeneous resultant into the resultant $R(y) = \mathrm{Res}_x(F, G)$ (by Property 7.3) which is a univariate polynomial in y (by Definition 4.1.2). Because the homogenisation ensures that the leading coefficient (in x) of $F_0^{(h)}$ and $G_0^{(h)}$ are non-zero constants, and the substitution $z = y + 1$ does not cause the highest degree term in the resultant (of degree $d_1 d_2$) to vanish, the resulting polynomial $R(y)$ retains this degree. Consequently,

$$R(y) = a_{d_1 d_2}\, y^{d_1 d_2} + \cdots + a_1\, y + a_0$$

with $a_{d_1 d_2} \neq 0$, thus ensuring $\deg\big(R(y)\big) = d_1 d_2$.

3) **Constant term plus shift is s:** In Step 1, the algorithm ensures that $F_0(x, 0)$ and $G_0(x, 0)$ are coprime in $\mathcal{F}[x]$. A standard property of resultants is that $\mathrm{Res}_x(p(x), q(x)) \neq 0$ if and only if $p(x)$ and $q(x)$ are coprime (see for example Theorem 3.18 in [56]). Evaluating the resultant $R(y)$ at $y = 0$ corresponds to calculating the resultant of $F(x, 0)$ and $G(x, 0)$. From Step 3, setting $y = 0$ means

the substitution $z = 1$ into the homogenised polynomials occurs as:

$$F(x, 0) = F_0^{(h)}(x, 0, 1) = F_0(x, 0),$$

and similarly $G(x, 0) = G_0(x, 0)$. Therefore,

$$R(0) = \mathrm{Res}_x(F(x, 0), G(x, 0)) = \mathrm{Res}_x(F_0(x, 0), G_0(x, 0)).$$

Since $F_0(x, 0)$ and $G_0(x, 0)$ were chosen to be coprime, their resultant $R(0)$ must be non-zero. Thus, the constant term $a_0 = R(0)$ is guaranteed to be non-zero.

Finally, Step 5 defines the public shift as $\delta_{\text{shift}} = s - a_0$, so

$$R(0) + \delta_{\text{shift}} = a_0 + (s - a_0) = s,$$

which establishes the desired relationship with the secret s.

Over a sufficiently large finite field $\mathcal{F}$, randomly chosen polynomials will satisfy the required conditions (i.e. coprimality at $y = 0$ and the specified degrees) with high probability. In the unlikely event that a chosen pair does not meet these conditions, the selection step can simply be repeated until a suitable pair is found.

Therefore, the algorithm is guaranteed to terminate successfully, returning the desired polynomials F and G, their resultant $R(y)$, and the shift δ_{shift}, which together possess the properties stated in the lemma. This completes the proof. □

Example 7.4.4. ***(Illustration of Algorithm 5)***

Consider the finite field $\mathcal{F}_{13}$, with degrees $d_1 = 2$ and $d_2 = 1$, and a secret $s = 11$. The aim is to find polynomials $F(x, y), G(x, y) \in \mathcal{F}_{13}[x, y]$ such that $\deg_x(F) = d_1$ and $\deg_x(G) = d_2$, with $\deg(\mathrm{Res}_x(F, G)) = d_1 d_2$, and a public shift δ_{shift} so that

$$\mathrm{Res}_x(F, G)(0) + \delta_{\text{shift}} \equiv 11 \pmod{13}.$$

Initially, random polynomials of the required degrees in x are selected. In this example, let the random polynomials F_0 and G_0 be:

$$F_0(x,y) = x^2 + (y+1)x + (2y+1), \quad G_0(x,y) = 2x + (3y+1).$$

These satisfy $\deg_x(F_0) = 2$ *and* $\deg_x(G_0) = 1$. *A check confirms that*

$$F_0(x,0) = x^2 + x + 1, \quad G_0(x,0) = 2x + 1$$

have no common factor in $\mathcal{F}_{13}[x]$ *(i.e. their GCD is 1).*

By introducing the variable z, each polynomial is homogenised:

$$\begin{aligned} F_0^{(h)}(x,y,z) &= x^2 + (y+z)x + 2yz + z^2, \\ G_0^{(h)}(x,y,z) &= 2x + (3y+z). \end{aligned}$$

Then, setting $z = y+1$ yields:

$$\begin{aligned} F(x,y) &= x^2 + (y + (y+1))x + 2y(y+1) + (y+1)^2 \\ &= x^2 + (2y+1)x + (3y^2 + 4y + 1), \\ G(x,y) &= 2x + 3y + (y+1) = 2x + (4y+1). \end{aligned} \tag{7.4}$$

For the resultant calculation $\mathrm{Res}_x(F,G)$*, the Sylvester matrix of F and G with respect to x is formed:*

$$\begin{vmatrix} 1 & 2y+1 & 3y^2+4y+1 \\ 2 & 4y+1 & 0 \\ 0 & 2 & 4y+1 \end{vmatrix}.$$

After performing row operations modulo 13 to obtain a triangular form, the resultant is found by multiplying the diagonal entries:

$$\mathrm{Res}_x(F,G) \equiv 12y^2 + 12y + 3 \pmod{13}.$$

Therefore, the resultant polynomial is

$$R(y) = 12y^2 + 12y + 3 \pmod{13}. \tag{7.5}$$

From this, $R(0) = 3$. *As the secret is* $s = 11$, *the shift is* $\delta_{\text{shift}} = s - R(0) = 11 - 3 = 8 \pmod{13}$.

Finally, the algorithm returns:

$$\begin{aligned} F(x, y) &= x^2 + (2y + 1)x + 3y^2 + 4y + 1, \\ G(x, y) &= 2x + (4y + 1), \\ R(y) &= 12y^2 + 12y + 3, \\ \delta_{\text{shift}} &= 8. \end{aligned} \tag{7.6}$$

These polynomials, together with the public shift δ_{shift}, *satisfy the requirements, therefore they are suitable for use in the secret sharing schemes.* □

This algorithm provides a reliable way to generate the required polynomials, forming a crucial part of the dealer's setup phase in the secret sharing schemes.

7.5 The Core Resultant-Based Secret Sharing Concept and Proposed Schemes

This section characterise the fundamental principles underlying the novel (t, n)-threshold secret sharing schemes that utilise polynomial resultants. As introduced, these schemes employ a *double-structured polynomial approach*, where the secret is embedded in the resultant of two bivariate polynomials, contrasting with traditional methods such as Shamir's scheme which uses a single univariate polynomial.

7.5.1 General Principle: Secret Encoding, Share Generation, and Reconstruction

The foundational concept involves the dealer performing the following steps:

1. **Secret Encoding:** The dealer selects a secret s. Using Algorithm 5 (Section 7.4.4), the dealer constructs two bivariate polynomials $F(x, y)$ and $G(x, y)$ over a (finite) field $\mathcal{F}$. These polynomials are specifically designed such that their resultant with respect to x, $R(y) = \text{Res}_x(F, G)$, is a polynomial in y of degree $t - 1$. The secret s is encoded in $R(y)$, typically as its constant term $R(0) = s$ (or $R(0) + \delta_{shift} = s$ for a known δ_{shift}).

2. **Share Generation:** The dealer generates n shares based on this algebraic framework. Shares can be numerical values derived from evaluations of $R(y)$, or symbolic information related to $F(x, y)$ and $G(x, y)$ at specific evaluation points. Distinct evaluation points $\alpha_1, \ldots, \alpha_n \in \mathcal{F}\backslash\{0\}$ are chosen for this purpose.

3. **Share Distribution:** Each participant P_i receives their respective share.

4. **Secret Reconstruction:** Any group of t or more participants can pool their shares. Using their t (or more) pieces of information, they can reconstruct the polynomial $R(y)$ via interpolation. Once $R(y)$ is known, the secret s is easily recovered (e.g. by evaluating $R(0)$).

5. **Security:** Any group of fewer than t participants should not be able to obtain any information about s. This relies on the fact that $t - 1$ points are insufficient to uniquely determine a polynomial of degree $t - 1$.

The novelty is in using the resultant of a bivariate system to define the main polynomial $R(y)$ that carries the secret, and in utilising properties of this resultant construction for enhanced features such as verifiability. Two distinct schemes based on this general principle are now detailed.

7.5.2 Approach 1: Dealer-Side Scaling

In this (t, n)-threshold approach, the dealer computes the resultant $R(y)$ and uses it as the basis to generate single-value shares $Sh_i = R(\alpha_i)$, which are distributed to the corresponding participants. The dealer also computes scaled evaluations V_i' and publishes binding commitments C_i to these values to facilitate verification. Additionally, a mechanism for identifying adversaries is provided, as detailed below.

A. Sharing Stage (Dealer):

(a) **Setup and Distribution:**

i. The dealer selects two bivariate polynomials $F(x, y)$ and $G(x, y)$ over a (finite) field $\mathcal{F}$, with $\deg_x(F) = d_1$ and $\deg_x(G) = d_2$. These polynomials are constructed (by employing Algorithm 5) so that their resultant $R(y) = \mathrm{Res}_x(F, G)$ has the precise degree $\deg_y(R(y)) = d_1 d_2 = t - 1$.

ii. The dealer computes the resultant of $F(x, y)$ and $G(x, y)$ with respect to x:

$$R(y) \;=\; \mathrm{Res}_x\big(F(x, y),\; G(x, y)\big).$$

(Note: This resultant $R(y)$ may be known in advance via Algorithm 5 as the target polynomial $R_{\text{target}}(y)$.)

iii. The secret s is then encoded using the relation

$$s \;=\; R(\beta) + \delta_{\text{shift}},$$

where β is a public evaluation point (typically $\beta = 0$) and the dealer also publishes the shift value δ_{shift}.

(b) **Public indices:** The dealer chooses *distinct* evaluation points

$\alpha_1, \ldots, \alpha_n \in \mathcal{F}$ and *publishes* the ordered sequence

$$(\alpha_1, \ldots, \alpha_n),$$

with the constraints $\alpha_i \neq \beta$, $\alpha_i \neq 0$, and $R(\alpha_i) \neq 0$. Since the α_i values are public and explicitly indexed, the interpolation coordinates used in reconstruction are fixed and cannot be replaced by alternative values without detection.

(c) **Private shares:** For each participant i ($i = 1, \ldots, n$), the dealer computes the share

$$Sh_i = R(\alpha_i)$$

and *privately* sends Sh_i to participant i.

(d) **Scaling:** The dealer chooses two random, non-zero elements $a, b \in \mathcal{F}$ and computes the scaled resultant

$$R'(y) \;=\; \mathrm{Res}_x\big(aF(x,y),\; bG(x,y)\big) \;=\; a^{d_2} b^{d_1} R(y).$$

Let $r = a^{d_2} b^{d_1}$ denote the scaling factor.

(e) **Commitments to scaled shares:** The dealer generates binding commitments. Let $\mathrm{Com}(\cdot;\tau)$ be a secure (binding, and preferably hiding) commitment scheme. For each i:

- compute the scaled evaluation $V_i' = R'(\alpha_i)$;
- sample a fresh random τ_i;
- compute the commitment

$$C_i \;=\; \mathrm{Com}\big(i, \alpha_i, V_i'; \tau_i\big).$$

No raw value V_i' is publicly published; instead, the dealer publishes the set of commitments $\{C_i\}_{i=1}^n$ so that participants can later verify the

integrity of their data. The actual values V_i' and τ_i are sent privately to the respective participants.

(f) The dealer may optionally publish the ratio value $r = a^{d_2} b^{d_1}$. (This is now safe with respect to share privacy, since each V_i' is hidden by the commitment.)

B. Verification and Complaint Stage (Participant i):

(a) **Commitment check:** Participant i verifies that the received opening is consistent with the public commitment:

$$C_i \stackrel{?}{=} \mathrm{Com}(i, \alpha_i, V_i'; \tau_i).$$

If this check fails, the participant broadcasts a complaint together with the evidence (V_i', τ_i).

(b) **Local ratio calculation:** Since $R(\alpha_i) \neq 0$ by construction, $Sh_i \neq 0$. Participant i computes the local ratio

$$\rho_i \;=\; \frac{V_i'}{Sh_i}.$$

For valid shares, ρ_i should coincide with the constant $r = a^{d_2} b^{d_1}$ across *all* honest participants.

(c) **Consistency check:** Participants employ one (or more) of the following techniques for consistency checking:

- *Dealer disclosure.* If the dealer publishes the scaling factor r, each participant i locally verifies that their computed ratio ρ_i equals r. Any mismatch directly indicates an issue with either the participant's share or the dealer's distribution.

- *Trusted combiner.* Each participant i securely sends their computed ratio ρ_i to a trusted combiner. The combiner checks whether

all received ρ_i values are identical. If a mismatch is detected, the combiner signals an inconsistency and typically identifies participant(s) j whose submitted ratio(s) ρ'_j differ from the others.

- *No combiner / hidden factor:* Participants engage in a secure multiparty computation (MPC) protocol (for example a private equality test), using their computed ρ_i values as inputs. The protocol determines whether all inputs are identical, without revealing the individual ρ_i values. An output of *false* signals an inconsistency. Identifying the specific adversary (or adversaries) then necessitates initiating the complaint stage.

- *Collaborative verification.* If the above individual methods are unavailable, at least t participants can collaborate:
  - **Step 1 (Filter):** The group verifies the commitments C_i for all submitted shares by checking that every (V'_i, τ_i) correctly opens C_i.
  - **Step 2 (Reconstruct R):** Using the public indices $(\alpha_1, \ldots, \alpha_n)$ and a subset of at least t valid shares $\{Sh_i\}$, the participants interpolate the polynomial

$$R^*(y) \text{ from } \{(\alpha_i, Sh_i)\}.$$

  - **Step 3 (Reconstruct R'):** From the corresponding scaled evaluations $\{V'_i\}$ (revealed by those participants), they interpolate a candidate

$$R'^*(y) \text{ from } \{(\alpha_i, V'_i)\}.$$

  - **Step 4 (Scaling-property check):** Participants then verify

whether there exists a field element ρ such that

$$R'^{*}(y) \;=\; \rho\, R^{*}(y).$$

If r is published, they additionally check $\rho \stackrel{?}{=} r$. Failure to satisfy this functional equality indicates a structural inconsistency or the presence of an adversary.

– **Step 5 (Global adversary detection):** If all checks in Steps 1–4 pass, the polynomials $R^{*}(y)$ and $R'^{*}(y)$ are treated as the recovered originals $R(y)$ and $R'(y)$. For *any* participant j (including those not in the initial collaborating subset), the group first checks any whose commitment opening has not yet been verified:

$$C_j \stackrel{?}{=} \mathrm{Com}(j, \alpha_j, V'_j; \tau_j).$$

They then test

$$Sh_j \stackrel{?}{=} R^{*}(\alpha_j), \qquad V'_j \stackrel{?}{=} R'^{*}(\alpha_j),$$

using the publicly known α_j and the privately held (Sh_j, V'_j, τ_j). Any mismatch identifies participant j as holding a corrupted or inconsistent share.

(d) **Complaint stage (if an inconsistency is found):**

i. If an inconsistency is detected (for example a commitment fails to open or the ratios mismatch), a complaint protocol is initiated.

ii. If a commitment check fails, the dealer is identified as inconsistent or shares are compromised (since commitments are binding).

iii. If a ratio or scaling check fails, the group excludes the inconsistent participant(s). To distinguish between a dealer fault (invalid

share) and a participant fault (altered share), the dealer may optionally participate by authenticating the original private distribution (for example via signatures).

C. Identifying Adversary:

(a) If a participant fails either the commitment check or the ratio/scaling check, they are identified as an adversary and excluded. If the dealer fails to satisfy the commitment binding, the dealer is identified as being at fault.

D. Secret Reconstruction:

(a) If all consistency checks pass, any subset of t or more shares can be used to interpolate $R(y)$ from the pairs (α_i, Sh_i). The secret is then computed as $s = R(\beta) + \delta_{\text{shift}}$ (typically with $\beta = 0$). The computational details of this reconstruction step are discussed in the subsequent subsection.

7.5.3 Secret Reconstruction

To recover the secret, any set $\mathcal{S}$ of t participants holding valid shares collaborate to perform polynomial interpolation. Each participant $i \in \mathcal{S}$ provides their verified private share $Sh_i = R(\alpha_i)$. From these t points (α_i, Sh_i), the polynomial $R(y)$ of degree $(t-1)$ is uniquely reconstructed using the Lagrange formula:

$$R(y) = \sum_{i \in \mathcal{S}} Sh_i \cdot \ell_i(y),$$

where $\ell_i(y)$ is the Lagrange basis polynomial:

$$\ell_i(y) = \prod_{\substack{j \in \mathcal{S} \\ j \neq i}} \frac{y - \alpha_j}{\alpha_i - \alpha_j}.$$

This polynomial satisfies $\ell_i(\alpha_i) = 1$ and $\ell_i(\alpha_j) = 0$ for $j \neq i$.

Finally, the secret s is recovered by evaluating $R(\beta)$ (typically at the public point $\beta = 0$) and adding the public shift δ_{shift}. Evaluating at $y = 0$ simplifies the Lagrange formula computation, as the basis polynomials $\ell_i(0)$ are calculated using only the public indices α_j:

$$R(0) = \sum_{i \in \mathcal{S}} Sh_i \cdot \ell_i(0).$$

This computed value $R(0)$, combined with the shift δ_{shift}, yields the secret:

$$s = R(0) + \delta_{\text{shift}}.$$

To facilitate implementation and ensure reproducibility, Algorithm 6 presents the Dealer-Side Scaling protocol in structured pseudocode. This algorithm formalises the precise sequence of operations required for the dealer, ranging from the construction of bivariate polynomials and the computation of the resultant to the determination of the public shift value δ_{shift}. It formally specifies the generation of participant shares, the calculation of scaled evaluations V_i', and the publication of binding commitments C_i, ensuring that the algebraic encapsulation of the secret is correctly established and verifiable prior to distribution.

Algorithm 6: Group Secret Reconstruction (Dealer-Side Scaling)

Input : Set of shares $\mathcal{S} = \{(\alpha_i, Sh_i, V_i', \tau_i)\}$ with $|\mathcal{S}| \geq t$
Public parameters: threshold t, evaluation point β, shift δ_{shift}, scaling factor r, and public commitments $\{C_i\}_{i\in\mathcal{S}}$
Output: Reconstructed secret $s \in \mathcal{F}$ (or $\perp$), Adversary set $\mathcal{A}$

`/* Step 1: Commitment Verification (Binding Check) */`
Initialise $\mathcal{A} := \emptyset$
foreach $i \in \mathcal{S}$ **do**
  if $C_i \neq \text{Com}(i, \alpha_i, V_i'; \tau_i)$ **then**
    $\mathcal{A} := \mathcal{A} \cup \{i\}$ `/* Dealer or participant fault */`
  end
end
$\mathcal{S} := \mathcal{S} \setminus \mathcal{A}$
if $|\mathcal{S}| < t$ **then**
  return $(\perp, \mathcal{A})$
end

`/* Step 2: Initial Interpolation (Candidate Generation) */`
Select subset $\mathcal{T} \subseteq \mathcal{S}$ such that $|\mathcal{T}| = t$
$R^*(y) := \text{LagrangeInterpolate}(\{(\alpha_i, Sh_i)\}_{i\in\mathcal{T}})$
$flag :=$ **false**
if $\exists j \in \mathcal{S} \setminus \mathcal{T}$ *such that* $R^*(\alpha_j) \neq Sh_j$ **then** $flag :=$ **true**

`/* Step 3: Verification via Scaling Ratio */`
foreach $i \in \mathcal{S}$ **do**
  if $Sh_i = 0$ **then return** $\perp$ `/* Invalid share as` $R(\alpha_i) \neq 0$ `by construction */`
  $\rho_i := V_i'/Sh_i$
end
if r *is public* **then**
  if $\exists i \in \mathcal{S}$ *such that* $\rho_i \neq r$ **then**
    $flag :=$ **true**
  end
else
  if *values* $\{\rho_i\}_{i\in\mathcal{S}}$ *are not identical* **then**
    $flag :=$ **true**
  end
  `/* Collaborative Interpolation Check */`
  $R'^*(y) := \text{LagrangeInterpolate}(\{(\alpha_i, V_i')\}_{i\in\mathcal{T}})$
  if $R'^*(y) \neq \rho_i \cdot R^*(y)$ **then**
    $flag :=$ **true**
  end
end

`/* Step 4: Adversary Identification and Subset Recovery */`
if $\deg(R^*) \geq t$ **or** $flag$ **then**
  Attempt to find a consistent subset $\mathcal{T}_{\text{valid}} \subseteq \mathcal{S}$ where $|\mathcal{T}_{\text{valid}}| = t$ and all ratio checks pass
  if *no such set exists* **then**
    return $(\perp, \mathcal{A})$
  else
    $R^*(y) := \text{LagrangeInterpolate}(\mathcal{T}_{\text{valid}})$
    foreach $j \in \mathcal{S}$ **do**
      if $R^*(\alpha_j) \neq Sh_j$ **or** $V_j'/Sh_j \neq r$ **then** $\mathcal{A} := \mathcal{A} \cup \{j\}$
    end
    return $(\perp, \mathcal{A})$
  end
end

`/* Step 5: Secret Extraction */`
$s := R^*(\beta) + \delta_{\text{shift}}$
return $s, \mathcal{A}$

7.5.4 Correctness of Dealer-Side Scaling and Illustrative Example

The correctness of Scheme 1 (Dealer-Side Scaling) is formally established in the following theorem.

Theorem 7.5.1 (Correctness of Dealer-Side Scaling)**.** *Let Scheme 1 (Dealer-Side Scaling) be performed over a (finite) field $\mathcal{F}$, with threshold t satisfying $t > \frac{n}{2}$. Assuming the dealer is honest and at least t participants hold valid shares, then:*

1. *Honest participants can detect inconsistent data (and hence invalid shares or openings) via the prescribed commitment and ratio/scaling consistency checks.*

2. *After excluding invalid participants, any collaborating subset of t honest participants can reconstruct the secret s.*

Proof. Let s be the secret that the dealer intends to share. The dealer applies Algorithm 5 to generate bivariate polynomials $F, G \in \mathcal{F}[x, y]$ with $\deg_x(F) = d_1$ and $\deg_x(G) = d_2$, ensuring $d_1 d_2 = t - 1$. By construction, the resultant

$$R(y) = \mathrm{Res}_x(F, G)$$

is a univariate polynomial of degree $(t - 1)$. The dealer fixes a public evaluation point β (typically $\beta = 0$) and publishes the shift

$$\delta_{\text{shift}} \;=\; s - R(\beta),$$

so that $s = R(\beta) + \delta_{\text{shift}}$.

The dealer then selects non-zero $a, b \in \mathcal{F}$, sets $r = a^{d_2} b^{d_1}$, and forms the scaled resultant

$$R'(y) = \mathrm{Res}_x(aF, bG) = r\, R(y).$$

For distinct public indices $\alpha_i \in \mathcal{F}$ with $\alpha_i \neq 0$, $\alpha_i \neq \beta$, and $R(\alpha_i) \neq 0$, the

dealer computes $Sh_i = R(\alpha_i)$ and privately distributes Sh_i to participant i. In addition, the dealer computes the scaled evaluations $V_i' = R'(\alpha_i)$ and publishes binding commitments

$$C_i = \mathrm{Com}(i, \alpha_i, V_i'; \tau_i),$$

while sending (V_i', τ_i) privately to participant i. The dealer may optionally publish r.

1) **Detecting inconsistent shares/openings:** Each participant i first verifies the commitment opening:

$$C_i \stackrel{?}{=} \mathrm{Com}(i, \alpha_i, V_i'; \tau_i).$$

If this check fails, the opening is inconsistent with the binding commitment and a complaint can be raised using the evidence (V_i', τ_i).

Assuming the commitment opens correctly, participant i computes the local ratio

$$\rho_i = \frac{V_i'}{Sh_i}, \qquad (Sh_i \neq 0),$$

which is well-defined under the public constraint $R(\alpha_i) \neq 0$ (hence $Sh_i \neq 0$). For valid data,

$$V_i' \;=\; R'(\alpha_i) \;=\; r\,R(\alpha_i) \;=\; r\,Sh_i,$$

and therefore $\rho_i = r$ for every honest participant. Any adversarial manipulation that causes either $Sh_i \neq R(\alpha_i)$ or $V_i' \neq R'(\alpha_i)$ (while still opening a commitment) will, except in the degenerate case where both are altered by the same factor, lead to $\rho_i \neq r$ and hence a detectable inconsistency. In practice, Scheme 1 employs one of the following ratio/scaling consistency mechanisms:

- *Dealer disclosure.* If the dealer publishes r, each participant checks whether $\rho_i = r$. Any mismatch flags inconsistency.

- *Trusted combiner.* If r is hidden but a trusted combiner is available, participants send ρ_i to the combiner, who checks whether all ratios coincide. Any discrepant ratio indicates an inconsistency attributable to the corresponding participant input.

- *No combiner / hidden factor.* Participants run an MPC procedure (e.g. private equality testing) to determine whether all ρ_i values are identical without revealing them. If the output is false, an inconsistency is detected and the complaint stage is initiated. Under the assumption $t > \frac{n}{2}$, the honest participants form a strict majority, enabling recovery of a consistent honest set for subsequent reconstruction.

- *Collaborative verification.* A subset $\mathcal{S}$ of size at least t reconstructs

$$R^*(y) \text{ from } \{(\alpha_i, Sh_i)\}_{i\in\mathcal{S}} \qquad \text{and} \qquad R'^*(y) \text{ from } \{(\alpha_i, V_i')\}_{i\in\mathcal{S}},$$

  and checks whether there exists $\rho \in \mathcal{F}$ such that $R'^*(y) = \rho\, R^*(y)$ (and, if r is public, additionally $\rho \stackrel{?}{=} r$). If $\mathcal{S}$ consists of honest participants, then necessarily $\rho = r$ since

$$\frac{V_i'}{Sh_i} = \frac{R'(\alpha_i)}{R(\alpha_i)} = r \qquad \text{for all } i \in \mathcal{S}.$$

  If $\mathcal{S}$ contains inconsistent inputs, then the scaling relation fails (or yields $\rho \neq r$ when r is known), thereby detecting inconsistency. By iterating over subsets, a consistent subset of honest participants can be identified; the strict-majority assumption $t > \frac{n}{2}$ guarantees the existence of such a subset of size t.

2) **Reconstructing the secret:** After excluding participants whose data are identified as inconsistent, assume at least t honest participants remain. Let $\mathcal{S} \subseteq$

$\{1, \ldots, n\}$ be a set of t participants holding valid pairs

$$\{(\alpha_i, R(\alpha_i)) \mid i \in \mathcal{S}\}.$$

Since R is a univariate polynomial of degree $(t-1)$, these t points uniquely determine $R(y)$ by the fundamental theorem of polynomial interpolation. The participants therefore recover $R(\beta)$ and compute

$$R(\beta) + \delta_{\text{shift}} = s,$$

which yields the intended secret. Hence any group of t honest participants can reconstruct s after excluding inconsistent participants, completing the correctness proof. □

The following example illustrates the scheme's operation.

Example 7.5.2. (Illustration of Scheme 1)

Let $\mathcal{F}_{13}$ be the field and suppose the secret is $s = 11$. Consider a (t, n)-threshold with $t = 3$ out of $n = 4$, taking $d_1 = 2$, $d_2 = 1$ (as $t - 1 = d_1 d_2 = 2$). Using Algorithm 5, the dealer selects the bivariate polynomials:

$$\begin{aligned} F(x, y) &= x^2 + (2y + 1)\, x \; + \; (3y^2 + 4y + 1), \\ G(x, y) &= 2\, x + (4y + 1). \end{aligned} \tag{7.7}$$

Their resultant w.r.t. x is

$$R(y) \;=\; 12y^2 + 12y + 3 \pmod{13}.$$

Since $R(0) = 3$, the shift is $\delta_{\text{shift}} = s - R(0) = 11 - 3 = 8 \pmod{13}$. The dealer publishes $\delta_{\text{shift}} = 8$. Then the dealer performs the following *setup* steps:

1. Picks non-zero $a = 3,\ b = 2$, and computes the scaled resultant

$$\begin{aligned} R'(y) &= a^{d_2}\, b^{d_1}\, R(y) \\ &= 3^1 \cdot 2^2 \cdot (12y^2 + 12y + 3) \\ &= 12(12y^2 + 12y + 3) \\ &\equiv y^2 + y + 10 \pmod{13}. \end{aligned}$$

   The scaling factor is $r = 12$.

2. Chooses distinct non-zero $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\} \subset \mathcal{F}_{13}$. For simplicity, let

$$\alpha_1 = 1,\ \alpha_2 = 2,\ \alpha_3 = 3,\ \alpha_4 = 4.$$

3. Computes each share as $Sh_i = R(\alpha_i)$ and its corresponding scaled evaluation as $V_i' = R'(\alpha_i)$, as shown in Table 7.1.

Table 7.1: Computed shares and scaled evaluations.

i	α_i	$Sh_i = R(\alpha_i)$	$V_i' = R'(\alpha_i)$
1	1	$R(1) = 1$	$R'(1) = 12$
2	2	$R(2) = 10$	$R'(2) = 3$
3	3	$R(3) = 4$	$R'(3) = 9$
4	4	$R(4) = 9$	$R'(4) = 4$

4. The dealer generates binding commitments $C_i = \mathrm{Com}(i, \alpha_i, V_i'; \tau_i)$ and publishes $\{C_i\}$. The values $(\alpha_i, Sh_i, V_i', \tau_i)$ are sent privately to participant i.

For the *verification* stage, participant i first verifies the commitment C_i and then computes

$$\rho_i = \frac{V_i'}{Sh_i} \pmod{13}.$$

For example: $\rho_1 = 12/1 \equiv 12 \pmod{13}$, $\rho_2 = 3/10 \equiv (3 \times 4) \equiv 12 \pmod{13}$.

If all shares are valid, each ratio should equal $r = 12 \pmod{13}$. Suppose that Participant 3 is an adversary and submits an incorrect share $Sh_3' = 5$ (instead of

4). The computed ratio would then be

$$\rho_3 = 9/5 = 9 \cdot 8 = 72 \equiv 7 \pmod{13},$$

which differs from the expected value of 12, thus identifying Participant 3 as an adversary.

The last stage of *complaint and reconstruction* includes:

1. The legitimate participants remove the adversary's share and use their pairs (α_1, Sh_1), (α_2, Sh_2), and (α_4, Sh_4) to interpolate $R(y)$.

2. For Lagrange interpolation at $y = 0$, the formula simplifies to

$$R(0) = Sh_1 \cdot \ell_1(0) + Sh_2 \cdot \ell_2(0) + Sh_4 \cdot \ell_4(0),$$

where each $\ell_i(0)$ is computed using the public indices:

$$\begin{aligned} R(0) &= 1 \cdot \frac{(-2)(-4)}{(1-2)(1-4)} + 10 \cdot \frac{(-1)(-4)}{(2-1)(2-4)} \\ &\quad + 9 \cdot \frac{(-1)(-2)}{(4-1)(4-2)} \\ &= \frac{8}{3} + 10 \cdot \frac{4}{-2} + 9 \cdot \frac{2}{6} \\ &\equiv 72 + 240 + 198 \pmod{13} \\ &\equiv 7 + 6 + 3 \equiv 16 \equiv 3 \pmod{13}. \end{aligned}$$

3. Thus the secret is recovered as $s = R(0) + \delta_{\text{shift}} = 3 + 8 = 11 \pmod{13}$.

This completes the reconstruction, after removing any invalid shares. □

Efficiency and Practicality: The computational load on each participant remains low, involving only standard cryptographic operations:

- Share verification: One commitment verification (typically a hash or exponentiation) followed by one field division to compute the ratio ρ_i.

- Secret reconstruction: Contribution of one share. The main work is polynomial interpolation, which is done collectively or by an authorised reconstructor.

The dealer performs the more intensive computations; generating F, G, computing $R(y) = \text{Res}_x(F, G)$, computing the scaled resultant $R'(y)$, evaluating Sh_i and V_i', and generating binding commitments C_i for all n participants. Regarding storage:

- Participant private storage: Each participant i stores their private share tuple $(\alpha_i, Sh_i, V_i', \tau_i)$.
- Public information: The dealer publishes the list of commitments $\{C_i\}_{i=1\ldots n}$, the public indices, and the public shift δ_{shift}.

This scheme is well-suited for scenarios where participant simplicity is a main priority and where a public broadcast channel is available for the dealer to publish commitments.

Remark 7.5.3. *The examples presented throughout this study are simplified for clarity and illustrative purposes. Practical cryptographic applications would require the use of substantially larger finite fields and potentially more complex polynomial structures.*

Remark 7.5.4. *As is standard practice in cryptography, the provided examples and proofs in this study disregard events occurring with negligible probability, such as an adversary accidentally generating data that mimics honest behaviour (for example, accidental or fabricated data that exactly matches a valid share and thus bypasses verification checks).*

Remark 7.5.5. *It is relevant to note that the Dealer-Side Scaling scheme, presented with two scaling factors a and b, can be readily specialised to a simpler variant. By setting the scaling factor $b = 1$, the overall scaling factor for the resultant becomes $r = a^{d_2}$, where $d_2 = \deg_x(G)$. In this specialised case, the dealer*

effectively chooses only one primary random scaling constant a to achieve the scaling of the resultant. This approach closely mirrors a scheme where the resultant $R(y)$ might be directly scaled by a single factor λ_R (where λ_R would represent this specialised overall scaling factor $r = a^{d_2}$). The underlying principles of share generation, verification through scaled values, and adversary detection remain fundamentally equivalent. The primary difference lies in the dealer's parameterisation of the scaling effect, with the $b = 1$ specialisation offering a simplification in the choice of scaling constants.

7.5.5 Approach 2: Participant-Side Scaling

In this (t, n)-threshold approach, the dealer provides the minimum information necessary for participants to reconstruct the secret through a multi-stage symbolic process. Unlike the direct distribution of resultant evaluations, the dealer privately issues masked symbolic coefficient shares. This allows a coalition of participants to first unmask the underlying bivariate polynomials $F(x, y)$ and $G(x, y)$ and subsequently compute the symbolic resultant $R(y)$ locally. To ensure the integrity of this process, the dealer publishes a structural anchor H_{root} and per-index scaling commitments C'_i, enabling robust adversary detection and verification as detailed below:

A. Sharing Stage (Dealer):

(a) **Setup and Symbolic Structure:**

i. The dealer selects two secret bivariate polynomials $F(x, y)$ and $G(x, y)$ over a finite field $\mathcal{F}$, with $\deg_x(F) = d_1$ and $\deg_x(G) = d_2$, and associated threshold t defined by $d_1 d_2 = t - 1$.

ii. The dealer computes the global symbolic resultant

$$R(y) \;=\; \mathrm{Res}_x\big(F(x, y),\, G(x, y)\big).$$

iii. The secret is defined as

$$s \;=\; R(\beta) + \delta_{\text{shift}},$$

where the evaluation point β (typically $\beta = 0$) and the shift δ_{shift} are public.

iv. **Post-quantum binding:** Assume a hiding, binding, post-quantum secure commitment scheme Com and a post-quantum secure hash function Hash. The dealer does not reveal any coefficients but forms private commitments $\mathcal{C}_F, \mathcal{C}_G$ to the coefficient vectors of F and G, and publishes only the compact and fixed-size anchor

$$H_{\text{root}} \;=\; \text{Hash}\big(\mathcal{C}_F \parallel \mathcal{C}_G\big),$$

thereby cryptographically binding to a single structural pair (F, G) without exposing the coefficients.

v. **Public scaling factor:** The dealer chooses non-zero $a, b \in \mathcal{F}^{\times}$ and defines

$$r \;=\; a^{d_2} b^{d_1}.$$

The value r (or a hash anchor $H(r) = \text{Hash}(r)$) is published for subsequent scaling checks.

(b) **Public indices:** The dealer publishes an ordered list of distinct evaluation points $\{\alpha_i\}_{i=1}^{n} \subset \mathcal{F}$, with $\alpha_i \neq \beta$, $R(\alpha_i) \neq 0$ and, if required by the application, $\alpha_i \neq 0$. These public indices fix the y-coordinates used in the reconstruction phase.

(c) **Secret-dependent values v_i:** For each participant i, the dealer computes the endpoint integrity tag

$$v_i \;=\; \alpha_i \, R(1) \, R(0).$$

(d) **Masked symbolic coefficient shares:** For every coefficient c of F and G, the dealer samples a random masking polynomial $h_c(z)$ of degree $t-1$ with $h_c(0) = c$. For participant i the masked share is

$$Sh_i \;=\; \{\, h_c(i) \,\}_{\text{all coeffs c}},$$

i.e. the collection of evaluations of all masking polynomials at $z = i$. The additional z-dimension provides an information–theoretic masking layer, ensuring that no single participant can recover any coefficient from their local data alone.

(e) **Scaling commitments:** Using the same a, b as above, the dealer computes

$$V_i' \;=\; r \cdot R(\alpha_i)$$

for each i. To bind these values, the dealer publishes a per-index commitment

$$C_i' \;=\; \mathrm{Com}\big(i, \alpha_i, V_i'; \tau_i\big),$$

where τ_i is fresh randomness. The raw values V_i' are not published.

(f) **Private distribution:** For each participant i, the dealer *privately* sends:

- the masked symbolic coefficient share Sh_i;
- the private commitment vectors $\mathcal{C}_F, \mathcal{C}_G$ (to be checked against H_{root});
- the verification metadata: v_i, the (possibly hashed) scaling factor r, and the opening (V_i', τ_i) to the public commitment C_i'.

B. Participant actions (private verification): Each participant i performs:

(a) **Structural binding check:** Verify that

$$\mathrm{Hash}\big(\mathcal{C}_F \parallel \mathcal{C}_G\big) \;=\; H_{\mathrm{root}},$$

ensuring that all participants are tied to the same global pair (F, G).

(b) **VSS and commitment checks:** Using the received $\mathcal{C}_F, \mathcal{C}_G$, participant i checks that their local masked share Sh_i is consistent with these commitments at the point $z = i$ (according to the chosen VSS instantiation), and verifies that the privately received (V_i', τ_i) correctly opens the public commitment C_i'.

C. Group verification and reconstruction:

(a) **Stage 1: Coefficient unmasking:** A coalition of at least t honest participants pools their shares $\{Sh_i\}$ and, for each coefficient, performs univariate interpolation in the masking variable z to recover

$$c \;=\; h_c(0),$$

thereby reconstructing the bivariate polynomials $F(x, y)$ and $G(x, y)$ from the unmasked coefficient set.

(b) **Stage 2: Resultant recovery and scaling filter:** The group computes

$$R^*(y) \;=\; \mathrm{Res}_x\big(F(x, y), G(x, y)\big)$$

from the reconstructed polynomials, and then checks for all participants in the coalition that

$$V_i' \stackrel{?}{=} r \cdot R^*(\alpha_i).$$

This scaling check ensures that the reconstructed resultant matches the dealer's committed per-index evaluations.

(c) **Stage 3: Endpoint integrity (v_i-check):** As an additional global consistency test, the group verifies for all participants i in the final coalition that

$$v_i \stackrel{?}{=} \alpha_i \, R^*(1) \, R^*(0).$$

This detects manipulations that might pass the scaling checks but alter the endpoint values $R^*(0)$ and $R^*(1)$, which are tied directly to the secret.

(d) **Secret recovery:** Once all checks pass, the secret is recovered as

$$s \;=\; R^*(0) + \delta_{\text{shift}}.$$

D. Identifying adversaries:

(a) **Structural failure:** If the H_{root} check or any VSS-consistency check on Sh_i fails, this indicates either a dishonest dealer or corrupted share distribution. In particular, a participant whose Sh_i fails verification (while others pass) is flagged as holding invalid data.

(b) **Algebraic / scaling failure:** If, after reconstruction, either

$$V_i' \neq r \, R^*(\alpha_i) \quad \text{or} \quad v_i \neq \alpha_i R^*(1) R^*(0)$$

for a given index i, then the corresponding participant is identified as providing inconsistent verification metadata (or as having tampered with their share), and is treated as adversarial.

Remark 7.5.6. *One may, at the implementation level, compress shares to univariate slices $F(x, \alpha_i)$ and $G(x, \alpha_i)$ derived from the masked symbolic coefficient shares, provided that the underlying VSS/commitment system supports suitable evaluation proofs against the structural anchor H_{root} and that this optimisation does not weaken the information-theoretic masking property of the z-dimension.*

From a computational perspective, Algorithm 7 formalises the participant-side protocols into a unified pseudocode. This algorithm details the systematic procedure for unmasking the bivariate coefficients via univariate interpolation in z, computing the symbolic resultant $R^*(y)$, and verifying consistency through the scaling filter and endpoint integrity tags (v_i). Furthermore, it explicitly defines the logic for isolating adversaries ($\mathcal{A}$) by cross-referencing local data against the reconstructed global polynomials, thereby ensuring the cryptographic soundness and resilience of the reconstruction process.

Algorithm 7: Participant-Side Scheme (Masked Symbolic Shares): Group Verification, Reconstruction, and Adversary Identification

Input : Collaborating set $\mathcal{S}$ with $|\mathcal{S}| \geq t$
For each $i \in \mathcal{S}$: masked symbolic coefficient share Sh_i, public index α_i, integrity tag v_i, and opening (V_i', τ_i) for the scaling commitment
Public parameters: threshold t, evaluation point β, shift δ_{shift}, scaling factor r, and public commitments $\{C_i'\}_{i\in\mathcal{S}}$
(Assume local VSS and anchor checks w.r.t. $\mathcal{C}_F, \mathcal{C}_G, H_{\text{root}}$ have already been performed by each participant.)
Output: Recovered secret $s \in \mathcal{F}$ (or $\perp$), Adversary set $\mathcal{A} \subseteq \mathcal{S}$

/* Step 1: Local Commitment Consistency (Scaling Anchors) */
$\mathcal{A} := \emptyset$
foreach $i \in \mathcal{S}$ **do**
  if $C_i' \neq \text{Com}(i, \alpha_i, V_i'; \tau_i)$ **then**
    $\mathcal{A} := \mathcal{A} \cup \{i\}$ /* Scaling commitment is inconsistent */
  end
end
if $\mathcal{A} \neq \emptyset$ **then**
  return $\perp$ /* Abort: Detected tampering with scaling data */
end
/* Step 2: Data Pooling and Pre-check */
Pool tuples $\{(\alpha_i, Sh_i, v_i, V_i')\}_{i\in\mathcal{S}}$
if $\exists i, j \in \mathcal{S}$ *s.t.* $\alpha_i = \alpha_j$ **or** $\alpha_i = \beta$ **or** $\alpha_i = 0$ **then**
  return $\perp$ /* Invalid identifiers detected */
end
/* Step 3: Two-Stage Resultant Reconstruction */
/* Stage 1: Unmask coefficients via univariate interpolation in z */
foreach *coefficient index* c **do**
  $c^* := \text{LagrangeInterpolate}(\{(i, Sh_{i,c})\}_{i\in\mathcal{S}})$ evaluated at $z = 0$
end
Construct bivariate polynomials $F^*(x, y)$ and $G^*(x, y)$ from $\{c^*\}$
/* Stage 2: Symbolic computation of the Resultant */
$R^*(y) := \text{Res}_x\big(F^*(x, y),\, G^*(x, y)\big)$
if $R^*(y) = \perp$ **then**
  return $\perp$ /* Insufficient or inconsistent shares for symbolic recovery */
end
/* Step 4: Scaling-Property Filter and v_i Verification */
foreach $i \in \mathcal{S}$ **do**
  /* Point-wise Scaling check: $V_i' \stackrel{?}{=} r \cdot R^*(\alpha_i)$ */
  if $V_i' \neq r \cdot R^*(\alpha_i)$ **then**
    $\mathcal{A} := \mathcal{A} \cup \{i\}$
  end
  /* Endpoint Integrity check: $v_i \stackrel{?}{=} \alpha_i R^*(1) R^*(0)$ */
  if $v_i \neq \alpha_i \cdot R^*(1) \cdot R^*(0)$ **then**
    $\mathcal{A} := \mathcal{A} \cup \{i\}$
  end
end
if $\mathcal{A} \neq \emptyset$ **then**
  /* If $|\mathcal{S} \setminus \mathcal{A}| \geq t$, re-run Step 3 with verified set; else abort. */
  if $|\mathcal{S} \setminus \mathcal{A}| < t$ **then**
    return $\perp$
  end
end
/* Step 5: Secret Extraction */
$s := R^*(\beta) + \delta_{\text{shift}}$
/* Step 6: Return Result */
return $s, \mathcal{A}$

Remark 7.5.7. *Note that the dealer privately sends the scaling factor $r = a^{d_2}b^{d_1}$ (or its hash $H(r)$) to each participant, while keeping the specific bivariate degrees d_1 and d_2 hidden. This masking of the exponents within the unified ratio r prevents an adversary from replicating the correct scaling filter for the resultant evaluations, thus further complicating any attempt to compromise the scheme. By hiding the individual degrees of the original polynomials, this approach provides a unique security advantage: participants can verify the algebraic consistency of the reconstructed resultant $R^*(y)$ against the committed values V_i' using only the public or privately received factor r. Consequently, the coalition can confirm the correctness of the structural unmasking without needing to determine the specific integer partition of (d_1, d_2) that constitutes the scaling factor.*

7.5.6 Correctness of Participant-Side Scaling and Illustrative Example

Similarly to Scheme 1, the correctness of Scheme 2 is proven as follows.

Theorem 7.5.8 (Correctness of Participant-Side Scaling with Masked Symbolic Shares)**.** *Let the Participant-Side Scaling Scheme be performed over a (finite) field $\mathcal{F}$. Assume that the dealer is honest and that at least t participants (with $t > \frac{n}{2}$) hold valid shares. Then:*

1. *Honest participants can detect invalid or tampered data via the structural binding check, the VSS/commitment checks, the scaling commitments, and the v_i endpoint checks.*

2. *After excluding invalid data, any collaborating subset of t honest participants can reconstruct the secret $s = R(0) + \delta_{shift}$.*

Proof. Let s be the secret the dealer intends to share. By Algorithm 5, the dealer generates bivariate polynomials

$$F(x, y),\, G(x, y) \in \mathcal{F}[x, y]$$

with $\deg_x(F) = d_1$, $\deg_x(G) = d_2$, and $d_1 d_2 = t - 1$. Algorithm 5 computes the resultant

$$R(y) = \mathrm{Res}_x(F, G),$$

which is guaranteed, by construction, to be a polynomial of degree $t - 1$ with non-zero constant term $R(0)$. The shift is defined as

$$\delta_{shift} = s - R(0),$$

and δ_{shift} is published.

The dealer then chooses distinct $\alpha_i \in \mathcal{F}$ (avoiding β and values for which $R(\alpha_i) = 0$), and random non-zero $a, b \in \mathcal{F}$. For each participant i, the dealer computes the integrity tag

$$v_i = \alpha_i \, R(1) \, R(0)$$

and the scaled evaluation

$$V_i' = r \, R(\alpha_i), \qquad r = a^{d_2} b^{d_1}.$$

A binding commitment $C_i' = \mathrm{Com}(i, \alpha_i, V_i'; \tau_i)$ is published, while V_i' and τ_i are sent privately. For each coefficient c of F and G, a masking polynomial $h_c(z)$ of degree exactly $t - 1$ with $h_c(0) = c$ is chosen, and the share of participant i is

$$Sh_i = \{h_c(i)\}_{\text{all coeffs}}.$$

The dealer forms private commitments $\mathcal{C}_F, \mathcal{C}_G$ to the coefficient vectors and publishes only the anchor

$$H_{\text{root}} = \mathrm{Hash}(\mathcal{C}_F \parallel \mathcal{C}_G).$$

We show that honest participants can detect invalid data and reconstruct s.

1) **Detecting invalid or tampered data.** Each participant i first performs a *structural binding check*:

$$\mathrm{Hash}(\mathcal{C}_F \parallel \mathcal{C}_G) \stackrel{?}{=} H_{\mathrm{root}}.$$

Since the commitment scheme Com and hash function Hash are binding, any discrepancy in the coefficient structure for different participants would change the hash and be detected.

Next, participant i verifies that their local masked share Sh_i is consistent with $\mathcal{C}_F, \mathcal{C}_G$ at $z = i$ (according to the chosen VSS instantiation). This ensures that the algebraic slice received by i matches the committed coefficient structure, without revealing the coefficients themselves. Since the dealer is honest, all honest participants pass this check.

Finally, participant i checks that the privately received (V_i', τ_i) opens the public commitment:

$$C_i' \stackrel{?}{=} \mathrm{Com}(i, \alpha_i, V_i'; \tau_i).$$

By the binding property of Com, the dealer cannot later change V_i', and the participant cannot claim a different value without causing the opening to fail.

Any failure in these checks for an index i implies that the data associated to that index is inconsistent, and the corresponding share is rejected.

2) **Secret reconstruction.** After excluding indices that fail the above checks, consider a collaborating subset $S \subseteq \{1, \ldots, n\}$ of size $|S| \geq t$ consisting of honest participants. For each coefficient c of F and G, the masking polynomial $h_c(z)$ has degree $t - 1$ and satisfies $h_c(0) = c$. Since at least t honest evaluations $\{h_c(i)\}_{i \in S}$ are available, univariate interpolation in z uniquely recovers h_c and hence $c = h_c(0)$. Therefore, the group reconstructs the original bivariate polynomials $F(x, y)$ and $G(x, y)$.

They then compute

$$R^*(y) = \mathrm{Res}_x(F, G).$$

By correctness of Algorithm 5 and the assumption of an honest dealer, it follows that $R^*(y) = R(y)$.

To confirm that the recovered $R^*(y)$ is consistent with the per-index commitments, the group checks, for each i in the accepted coalition,

$$V_i' \stackrel{?}{=} r\, R^*(\alpha_i).$$

Since $V_i' = rR(\alpha_i)$ was computed by the honest dealer and $R^*(y) = R(y)$, this equality holds for all honest participants. Any mismatch would indicate either a tampered share or incorrect interpolation, and the corresponding index is rejected.

As a further endpoint integrity check, the group computes $R^*(0)$ and $R^*(1)$ and verifies, for all i in the final coalition,

$$v_i \stackrel{?}{=} \alpha_i\, R^*(1)\, R^*(0).$$

Again, for the honest dealer and correctly reconstructed R^*, this equality holds, since v_i was originally defined using $R(0)$ and $R(1)$. Any discrepancy reveals inconsistent verification metadata or tampered shares.

Once all accepted indices pass both the scaling check and the v_i check, the coalition has confirmed that $R^*(y)$ is indeed the original resultant $R(y)$. The secret is then recovered as

$$s \;=\; R^*(0) + \delta_{shift} \;=\; R(0) + \delta_{shift}.$$

Thus, under the assumptions of an honest dealer and at least t honest participants, any adversarial modification of shares or verification values is detected via the commitment, scaling, or v_i checks, and a subset of t honest participants can reconstruct $R(y)$ and therefore the correct secret s. □

An illustrative example demonstrating its practical operation follows:

Example 7.5.9. (Illustration of Participant-Side Scheme with Masked Symbolic Shares)

This example revisits Scheme 2 using the same bivariate polynomials (see Equations 7.4) generated by Algorithm 5, and previously used in Example 7.5.2:

$$\begin{aligned} F(x,y) &= x^2 + (2y+1)\,x \;+\; (3y^2+4y+1), \\ G(x,y) &= 2\,x + (4y+1). \end{aligned} \tag{7.8}$$

Over $\mathcal{F}_{13}$, their resultant with respect to x is

$$R(y) \;=\; 12y^2 + 12y + 3$$

(see Equation 7.5). Suppose the secret is $s = 11$. Since $R(0) = 3$, the dealer sets the public shift

$$\delta_{shift} \;=\; s - R(0) \;\equiv\; 11 - 3 \;\equiv\; 8 \pmod{13}.$$

Sharing stage (dealer).

1. The dealer confirms $d_1 = \deg_x(F) = 2$ and $d_2 = \deg_x(G) = 1$, so $d_1 d_2 = 2 \cdot 1 = 2 = t - 1$ and hence the threshold is $t = 3$ out of $n = 4$.

2. The dealer chooses distinct non-zero

$$\alpha_1 = 1, \quad \alpha_2 = 2, \quad \alpha_3 = 3, \quad \alpha_4 = 4$$

   in $\mathcal{F}_{13}$, none of which makes $R(\alpha_i)$ vanish.

3. The dealer computes

$$R(1) = 12 + 12 + 3 = 27 \equiv 1 \pmod{13}, \qquad R(0) = 3,$$

and defines, for each $i = 1, \ldots, 4$, the secret-dependent tags

$$v_i \;=\; \alpha_i \, R(1)\, R(0) \;=\; 3\alpha_i \pmod{13}.$$

Thus

$$v_1 = 3, \quad v_2 = 6, \quad v_3 = 9, \quad v_4 = 12.$$

4. The dealer selects random $a = 4$ and $b = 2$ in $\mathcal{F}_{13}$ and computes the global scaling factor

$$r = a^{d_2} b^{d_1} = 4^1 \cdot 2^2 = 4 \cdot 4 = 16 \equiv 3 \pmod{13},$$

which will be used in the scaling check.

5. **Masked symbolic coefficient shares.** For each scalar coefficient c occurring in F and G, the dealer chooses a masking polynomial $h_c(z) \in \mathcal{F}_{13}[z]$ of degree exactly $t - 1 = 2$ with $h_c(0) = c$. For illustration, consider the coefficient of x^2 in F, namely $c = 1$. The dealer might choose, for instance,

$$h_{x^2}(z) = 1 + 2z + z^2 \;\in\; \mathcal{F}_{13}[z],$$

which satisfies $h_{x^2}(0) = 1$. For participant i this yields the masked value $h_{x^2}(i)$; the other coefficients are treated analogously with their own degree-2 masking polynomials. The complete masked share of participant i is then the vector

$$Sh_i = \{h_c(i)\}_{\text{all scalar coefficients } c \text{ of } F,G}.$$

These values do *not* reveal the coefficients c.

6. **Scaling commitments.** The dealer evaluates the global resultant at each

public point:

$$R(1) = 1,$$
$$R(2) = 12 \cdot 4 + 12 \cdot 2 + 3 = 48 + 24 + 3 = 75 \equiv 10 \pmod{13},$$
$$R(3) = 12 \cdot 9 + 12 \cdot 3 + 3 = 108 + 36 + 3 = 147 \equiv 4 \pmod{13},$$
$$R(4) = 12 \cdot 16 + 12 \cdot 4 + 3 = 192 + 48 + 3 = 243 \equiv 9 \pmod{13}.$$

Using $r = 3$, the scaled evaluations are

$$V_i' = r\, R(\alpha_i) \quad \Rightarrow \quad V_1' = 3,\ V_2' = 4,\ V_3' = 12,\ V_4' = 1.$$

For each i, the dealer publishes a binding commitment

$$C_i' = \mathrm{Com}(i, \alpha_i, V_i'; \tau_i),$$

but keeps V_i' and τ_i private.

Table 7.2 summarises the symbolic data points (α_i, V_i', v_i) held by participants. Note that Sh_i represents the n-tuple of masked coefficients.

Table 7.2: Computed values for each participant in the scheme.

i	Sh_i (Sample masked G coeff)	α_i	V_i' (Scaled)	v_i
1	$\ldots, 4, \ldots$	1	$3 \cdot R(1) = 3$	3
2	$\ldots, 9, \ldots$	2	$3 \cdot R(2) = 30 \equiv 4$	6
3	$\ldots, 3, \ldots$	3	$3 \cdot R(3) = 12$	9
4	$\ldots, 12, \ldots$	4	$3 \cdot R(4) = 27 \equiv 1$	12

7. **Private distribution.** To participant i, the dealer privately sends:

$$Sh_i, \quad v_i, \quad r, \quad \mathcal{C}_F, \mathcal{C}_G, \quad (V_i', \tau_i),$$

together with the global anchor $H_{\mathrm{root}} = \mathrm{Hash}(\mathcal{C}_F \parallel \mathcal{C}_G)$ (the latter may be public).

Participant-side verification.

Each participant i performs:

- *Structural binding check:*

$$\mathrm{Hash}(\mathcal{C}_F \parallel \mathcal{C}_G) \stackrel{?}{=} H_{\mathrm{root}}.$$

- *VSS consistency check:* verify that the masked vector Sh_i is consistent with $\mathcal{C}_F, \mathcal{C}_G$ when evaluated at $z = i$.

- *Commitment check:* verify that (V_i', τ_i) opens the public commitment:

$$C_i' \stackrel{?}{=} \mathrm{Com}(i, \alpha_i, V_i'; \tau_i).$$

If all checks pass, participant i stores (Sh_i, v_i, V_i') as a valid share. Note that in the *actual* protocol the participants do not know F, G, $R(\alpha_i)$ or the underlying coefficients; these explicit values are shown only here for illustration.

Group reconstruction and checks.

Suppose participants $1, 2$ and 4 are honest and form a coalition of size $|S| = 3 = t$. They proceed as follows.

1. *Stage 1: Coefficient unmasking.* For each coefficient c of F and G, the values $\{h_c(i)\}_{i \in S}$ give three evaluations of the degree-2 polynomial $h_c(z)$. Univariate interpolation in z (over $\mathcal{F}_{13}$) uniquely recovers $h_c(z)$, and hence $c = h_c(0)$. Repeating this for all coefficients yields the original polynomials $F(x, y)$ and $G(x, y)$ as above. In this case, for the constant term of G, they interpolate $(1, 4), (2, 9), (4, 12)$ to find $h_1(0) = 1$.

2. *Stage 2: Resultant and scaling filter.* From the recovered F and G, the coalition computes

$$R^*(y) = \mathrm{Res}_x(F, G) = 12y^2 + 12y + 3.$$

They then verify, for each $i \in \{1, 2, 4\}$, that the scaled evaluations are consistent:

$$V_i' \stackrel{?}{=} r\, R^*(\alpha_i).$$

Using the values already computed,

$$\begin{aligned} rR^*(\alpha_1) &= 3 \cdot 1 = 3 = V_1', \\ rR^*(\alpha_2) &= 3 \cdot 10 = 30 \equiv 4 = V_2', \\ rR^*(\alpha_4) &= 3 \cdot 9 = 27 \equiv 1 = V_4', \end{aligned}$$

so the scaling check passes for this coalition.

3. *Stage 3: Endpoint integrity via v_i-check.* The coalition computes $R^*(0) = 3$ and $R^*(1) = 1$ and verifies

$$v_i \stackrel{?}{=} \alpha_i\, R^*(1)\, R^*(0) = 3\alpha_i$$

for $i = 1, 2, 4$. Using the values above:

$$v_1 = 3, \quad v_2 = 6, \quad v_4 = 12,$$

so the check holds for all three indices. Any participant providing inconsistent v_i would be detected at this step.

4. *Secret recovery.* Since $R^*(y) = R(y)$ has been validated, the secret is recovered as

$$s \;=\; R^*(0) + \delta_{shift} \;\equiv\; 3 + 8 \;\equiv\; 11 \pmod{13}.$$

In this way the example shows, using the same polynomials F, G as in the previous example, how the participant-side scheme combines masked symbolic shares, scaling-based commitments, and the v_i endpoint tags to provide verifiable reconstruction while keeping the underlying coefficients hidden from all individual participants. □

This scheme is suited for scenarios where higher security assurance and effective participant-driven verification are prioritised over minimal participant computation, and where the potential for post-quantum resistance is especially desired.

7.5.7 Underlying Algebraic Foundations

Both Scheme 1 (Dealer-Side Scaling) and Scheme 2 (Participant-Side Scaling) fundamentally rely on the algebraic properties of polynomial resultants discussed in Section 7.2.1.

- The *evaluation property* is central to verification and reconstruction. In Scheme 1, participants use scalar shares $Sh_i = R(\alpha_i)$ for interpolation of $R(y)$. In Scheme 2, the public indices α_i determine the y-coordinates at which the reconstructed resultant $R^*(y)$ is checked against the dealer's committed scaled values V_i' (and against the endpoint tag v_i), whilst reconstruction proceeds by unmasking coefficients rather than directly interpolating $R(y)$.

- The *scaling property* underpins verifiability in both schemes. In Scheme 1, the dealer scales the input polynomials by a and b, inducing a global factor $r = a^{d_2}b^{d_1}$ on the resultant, so that participants can verify consistency via the ratios $\rho_i = V_i'/Sh_i$. In Scheme 2, the dealer likewise defines $r = a^{d_2}b^{d_1}$ and commits per index to $V_i' = r \cdot R(\alpha_i)$; after the coalition reconstructs F and G and computes $R^*(y) = \mathrm{Res}_x(F(x,y), G(x,y))$, correctness is verified by checking the global relation $V_i' \stackrel{?}{=} r \cdot R^*(\alpha_i)$ (and, in addition, by the endpoint integrity check $v_i \stackrel{?}{=} \alpha_i R^*(1)R^*(0)$).

- The *polynomial combination property* and the resultant's relationship to common roots ensure that $R(y)$ captures the secret as embedded by the dealer through $F(x,y)$ and $G(x,y)$. Algorithm 5 is designed to ensure that

$R(y)$ has the desired structure (degree $t-1$) and supports the chosen secret-encoding relation $s = R(\beta) + \delta_{\text{shift}}$.

The formal correctness of Algorithm 5 is established by analysing its algebraic construction steps, confirming that it produces polynomials F and G whose resultant $R(y)$ meets the specified degree and secret-encoding requirement.

The formal proofs for Scheme 1 (Theorem 7.5.1) and Scheme 2 (Theorem 7.5.8) establish their (t, n)-threshold properties (correct reconstruction by at least t participants, and privacy against any coalition of fewer than t participants), as well as the effectiveness of their respective verification and adversary identification techniques. These proofs rely on polynomial interpolation theory (for correctness and privacy) together with algebraic properties of resultants (for the scaling-based and structural verification aspects). For Scheme 1, privacy follows from the fact that fewer than t evaluations of a degree-$(t-1)$ polynomial do not determine $R(\beta)$. For Scheme 2, privacy is enforced by the information–theoretic masking in the auxiliary variable z, so that fewer than t masked coefficient evaluations do not reveal any coefficient of F or G, and hence do not reveal $R(\beta)$. Verifiability in both schemes relies on the uniqueness of the resultant and its scaled counterparts under the prescribed construction and commitment mechanisms. The arguments presented within the scheme descriptions in this chapter outline the main reasoning for their security and functionality.

7.6 Comparative Analysis of the Proposed Secret Sharing Schemes

The two proposed schemes, Dealer-Side Scaling (Scheme 1) and Participant-Side Scaling (Scheme 2), offer distinct operational characteristics and trade-offs. A comparative analysis is therefore essential for selecting the appropriate scheme based on specific application requirements, security considerations, and resource constraints.

7.6.1 Flowcharts for Comparing Scheme Approaches

The flowcharts in Figure 7.1 offer a side-by-side visual comparison of the two schemes discussed throughout this study. These diagrams outline the key stages for each approach, including initial setup, the differing methods of share distribution and computation, the distinct verification procedures (local ratio ρ_i checks for Scheme 1 versus scaling and endpoint integrity checks for Scheme 2), and the final secret reconstruction, thus clarifying the allocation of responsibilities between the dealer and participants.

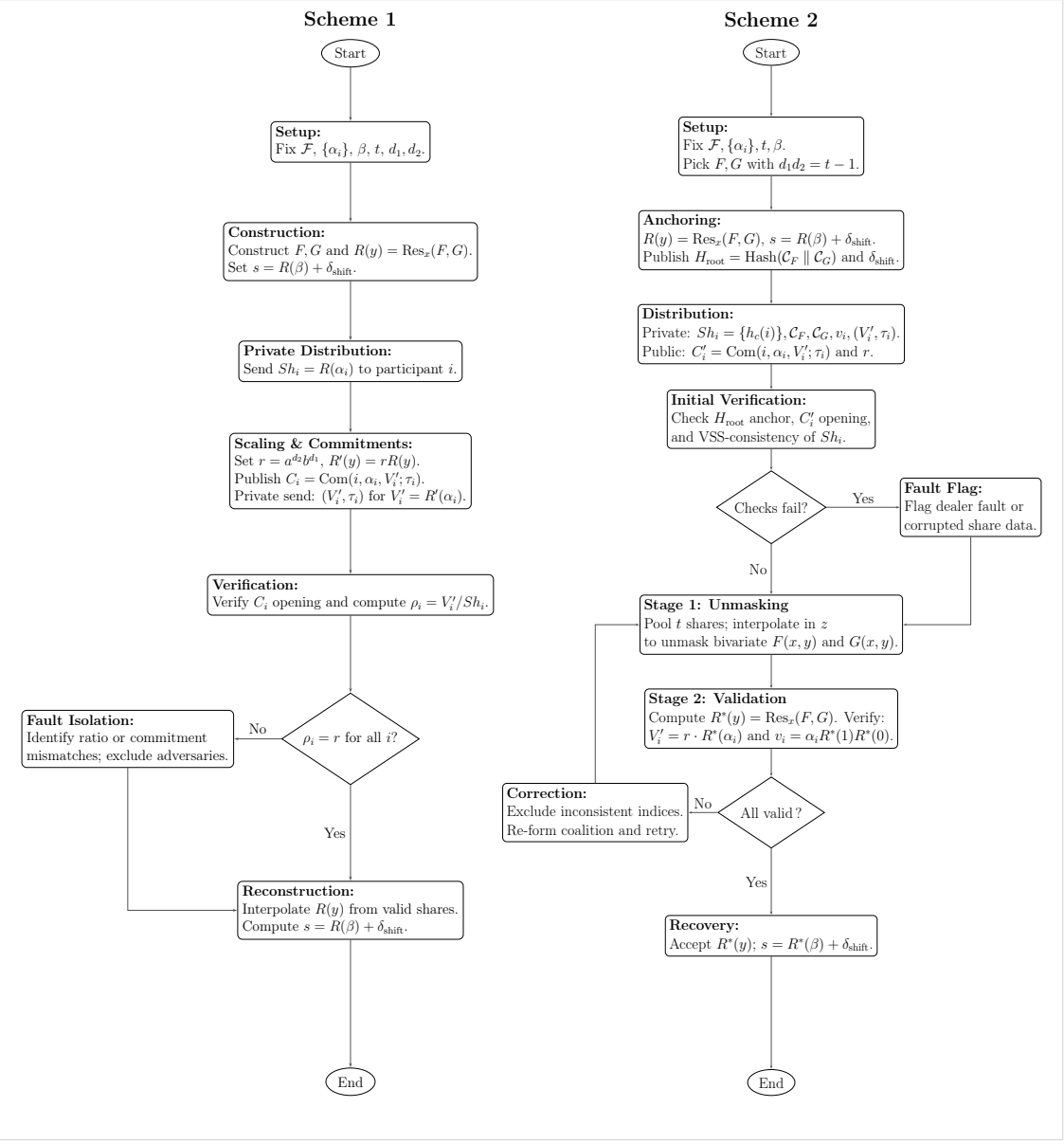

Figure 7.1: Comparative flowcharts for Dealer-Side (left) and Participant-Side (right) scaling processes.

7.6.2 Key Comparative Features

This section evaluates the two resultant-based secret sharing schemes developed in this chapter: the *Dealer-Side Scaling* scheme (Scheme 1) and the *Participant-Side Scaling* scheme (Scheme 2). The analysis is structured around key operational and security characteristics to highlight their distinct approaches and practical trade-offs.

- **Nature of Shares:**
  - *Scheme 1 (Dealer-Side Scaling):* Distributes *numerical shares* $Sh_i = R(\alpha_i) \in \mathcal{F}$, i.e. single field elements tied to public indices.
  - *Scheme 2 (Participant-Side Scaling):* Distributes *masked symbolic coefficient shares* $Sh_i = \{h_c(i)\}_{\text{all coeffs } c}$ that collectively enable reconstruction of the bivariate polynomials $F(x, y)$ and $G(x, y)$ (rather than directly distributing $R(\alpha_i)$ or univariate slices).
- **Participant Computational Load:**
  - *Scheme 1:* Minimal. Verification involves checking a commitment opening (if used), computing a single ratio $\rho_i = V_i'/Sh_i$, and comparing it either to a public r or for equality across participants. Reconstruction requires contributing one field element.
  - *Scheme 2:* Higher. Each participant performs private verification steps (structural binding via H_{root}, VSS-consistency of masked shares, and commitment opening checks). Group reconstruction requires coefficient unmasking via interpolation in the masking variable z, followed by a resultant computation and global checks (scaling and endpoint integrity).
- **Dealer's Computational Load:**
  - *Scheme 1:* Computes $F, G, R(y)$ and the scaled resultant $R'(y) = rR(y)$. It then performs n evaluations $Sh_i = R(\alpha_i)$ and n evaluations

$V_i' = R'(\alpha_i)$, together with generating and publishing commitments to the V_i' values.

- *Scheme 2:* Computes $F, G, R(y)$, selects a, b and publishes r (or a hash anchor), forms coefficient commitments (publishing only H_{root}), generates masking polynomials for each coefficient, and distributes the masked coefficient evaluations Sh_i. It also computes per-index values $v_i = \alpha_i R(1)R(0)$ and publishes commitments C_i' to $V_i' = r \cdot R(\alpha_i)$.

- **Storage Requirements for Participants:**
  - *Scheme 1:* Minimal. Stores Sh_i and, for verification, the opening information (V_i', τ_i).
  - *Scheme 2:* Higher. Stores the masked coefficient share $Sh_i = \{h_c(i)\}$, the commitment vectors $\mathcal{C}_F, \mathcal{C}_G$ (or the material required to validate them against H_{root}), and the verification metadata (v_i, V_i', τ_i) (and r or $H(r)$).

- **Communication Overhead (during reconstruction):**
  - *Scheme 1:* Minimal. Each participant submits (α_i, Sh_i) (and, if required for global checking, the opening (V_i', τ_i)) to enable interpolation and consistency validation.
  - *Scheme 2:* Higher. Participants pool their masked coefficient shares $\{Sh_i\}$ to enable coefficient unmasking; the coalition then uses the privately held openings (V_i', τ_i) and tags v_i for the global scaling and endpoint integrity checks.

- **Verification Mechanism and Point of Detection:**
  - *Scheme 1:* Dealer-assisted verification. Participants (or a reconstructor) verify shares by checking commitment openings and computing $\rho_i = V_i'/Sh_i$. Consistency is checked either by comparing ρ_i to a

published r, or by ensuring equality of the ratios across collaborating participants.

  - *Scheme 2:* Multi-layered, participant-centric verification. Participants first validate a global structural binding via H_{root} and verify VSS/commitment consistency of their masked data. After reconstruction, the coalition verifies $V_i' \stackrel{?}{=} r \cdot R^*(\alpha_i)$ and $v_i \stackrel{?}{=} \alpha_i R^*(1)R^*(0)$, providing both algebraic and endpoint integrity assurance.

- **Security Characteristics and Effectiveness:**

  - *Scheme 1:* Provides verifiability via binding commitments to scaled evaluations and a simple scaling relation, making it effective at detecting inconsistent scalar shares.

  - *Scheme 2:* Provides stronger structural assurance by combining information–theoretic masking of coefficients with cryptographic binding (H_{root} and commitments to V_i'), and by enforcing both scaling consistency and endpoint integrity via the v_i check. This makes manipulation significantly harder, and the scheme is designed to be compatible with post-quantum secure commitment and hashing primitives.

7.6.3 Comparison with Established Threshold Schemes

To contextualise the two resultant-based schemes within the established literature, Table 7.3 provides a comparative analysis of their key characteristics. The comparison evaluates each scheme across several features, including share structure, computational costs, and the nature of their verification and security properties. The features are listed row-wise to facilitate a direct assessment across the four schemes.

Table 7.3: Comparative analysis of secret–sharing schemes.

Feature	Shamir [120]	Liu et al. [96]	Dealer-Side Scaling	Participant-Side Scaling
Share Size	1 field element	2 field elements	1 element + proof	Masked set + tags
Share Type	Numeric point	Numeric point + tag	Numeric + Comm.	Masked Symbolic
Dealer Complexity	Low (polynomial generation)	Moderate (bivariate poly + tag generation)	Moderate (Resultant + Commitments)	High (Masking + Anchors)
Participant Complexity	Low (polynomial interp.)	Low (polynomial interp.)	Low (Commit. check + ratio)	Moderate (Unmasking + Resultant)
Verify Type	Add-on (external commitments)	Built-in tags (limited)	Binding Comm. + Ratio Check	Structural (H_{root}) + Algebraic
Dealer Dependence	Requires dealer / third-party commitments	Fully dealer generated tags	Requires dealer's binding commitment	Requires dealer's structural anchor
Adversary Detection	None (only detects reconstruction failure)	Signals inconsistency but cannot localise	Pinpoints invalid shares globally	Pinpoints faults (Dealer/Part.)
Dealer Trust	Honest-dealer assumption critical	Honest-dealer assumption critical	Verifiable (detects inconsistency)	Verifiable. Anchor prevents structural switching.
Authentication Source	External or dealer commitments	Dealer publishes tags with shares	Binding Comm. + Scaling Factor	Root Hash + Integrity Tag (v_i)
Post-Quantum Safety	Classical finite-field security	Classical hash/tag security	High (PQ Commit + Hashing)	High. Symbolic masking + PQ Anchors

7.6.4 Summary of Trade-offs

The choice between Dealer-Side Scaling (Scheme 1) and Participant-Side Scaling (Scheme 2) involves clear trade-offs:

- **Scheme 1** offers simplicity and efficiency for participants, making it well-suited for scenarios involving resource-constrained participants such as in *IoT devices*, *large-scale networks*, or applications where rapid share processing is required such as in *ephemeral key reconstruction in secure messaging protocols*. Its security relies on the integrity of the dealer's binding commitments and publicly provided verification data.

- **Scheme 2** provides a stronger, participant-driven verification process during reconstruction, offering greater resistance against sophisticated structural tampering. The use of masked symbolic shares and private scaling filters makes the verification process more effective but increases computational load, storage, and communication for participants. Additionally, it shows potential for enhanced post-quantum security due to its information-theoretic masking layer, making it particularly well-suited for high-assurance applications where strong verification is critical, such as in the management of *cryptocurrency multi-signature wallets* or the protection of *national security communications*.

The following table provides a concise comparative analysis of the two proposed schemes, designed to facilitate a direct assessment of their respective strengths, trade-offs, and ideal application scenarios.

Table 7.4: Comparison of Dealer-Side Scaling and Participant-Side Scaling Approaches

Feature	**Dealer-Side Scaling**	**Participant-Side Scaling**
Share Distribution	Dealer sends numerical shares $R(\alpha_i)$ (single values) + private opening data.	Dealer sends Masked Symbolic Coefficient Shares $Sh_i = \{h_c(i)\}$ (sets of values).
Scaling Ratio	Computed by the dealer as part of the scaled resultant $R'(y) = rR(y)$.	Used by participants as an algebraic filter $V_i' = rR^*(\alpha_i)$ on the reconstructed resultant.
Local Self Verification	Achievable via local verification of $\rho_i = r$ (if r is public) and commitment checks C_i.	Participants verify structural anchor H_{root} locally; algebraic verification requires coalition unmasking.
Adversary Screening	Dealer fault detected early via commitment check. Participant fault detected via ratio ρ_i.	Dealer fault detected via H_{root} mismatch. Participant fault detected via algebraic filter (V_i') mismatch.
Verification Process	Participants check C_i opening, then compute $\rho_i = V_i'/Sh_i$. Consistency requires $\rho_i = r$.	Participants unmask coefficients, compute $R^*(y)$, and verify $V_i' = rR^*(\alpha_i)$ and endpoint tags v_i.
Complaint Stage	Commitment failure or ratio ρ_i mismatch initiates complaint.	Structural hash mismatch or algebraic filter failure initiates iterative subset verification.

Continued on next page

– Comparison of Scaling Approaches (Continued) –

Feature	Dealer-Side Scaling	Participant-Side Scaling
Adversary Handling	Adversary identified by failed commitment or ratio $\rho_i \neq r$.	Adversary identified by inconsistency with global resultant $R^*(y)$ or structural anchor.
Verification Reliant	Relies on dealer's binding commitments and scaled data integrity.	Relies on structural binding (H_{root}) and collaborative algebraic consistency.
Computational Load	Dealer-heavy ($2n$ evals + commitments). Participant-light (hash + ratio check).	Participant-heavy (unmasking interpolation + local symbolic resultant computation).
Data Leakage	Minimal: participants only receive numerical share values, $R(\alpha_i)$.	Minimal: participants receive masked vectors; coefficients remain information-theoretically hidden.
Network Suitability	Suitable for large networks due to minimal participant-side computation.	Suitable for small to medium networks due to reconstruction complexity.
Example Applications	Large-scale IoT networks, time-sensitive protocols, or resource-constrained nodes.	High-assurance contexts (e.g., multi-sig wallets, national security) requiring robust structure.
Post-Quantum Safety	Compatible with post-quantum commitments/hashes; algebraic core does not rely on DL/factoring.	Potentially PQ secure due to symbolic masking structure + PQ anchors.

Ultimately, the selection depends on the specific security goals, the threat model, the computational capabilities of the participants, and the acceptable level of complexity for the system. Both schemes demonstrate the adaptability of using polynomial resultants as a foundation for constructing feature-rich secret sharing protocols.

Chapter Summary

This chapter has introduced a novel framework for (t, n)-threshold secret sharing by utilising the algebraic theory of polynomial resultants. This investigation has demonstrated not only the theoretical feasibility but also the practical advantages of constructing secret sharing schemes based on the resultant of two bivariate polynomials. This approach, to the best of current knowledge, represents a new avenue in cryptographic scheme design, moving beyond traditional single-polynomial constructions such as Shamir's.

Two distinct secret sharing schemes have been developed and presented:

1. **Scheme 1 (Dealer-Side Scaling):** This scheme prioritises participant efficiency by distributing simple numerical shares. Verification is facilitated by the dealer who provides binding commitments to scaled evaluations, enabling participants to cryptographically verify their shares and allowing the reconstruction group to validate submitted shares against public data.

2. **Scheme 2 (Participant-Side Scaling):** This scheme employs masked symbolic coefficient shares and empowers participants with structural anchors and private scaling filters to engage in a more active, multi-layered verification process during reconstruction. This "double-structured polynomial approach" offers potentially enhanced security effectiveness and is designed to accommodate post-quantum secure commitment and hashing primitives.

A critical enabler for these schemes is Algorithm 5, a method for constructing the input bivariate polynomials $F(x, y)$ and $G(x, y)$ such that their resultant $R(y) = \text{Res}_x(F, G)$ has the exact degree $t - 1$ required for a (t, n)-threshold and correctly encodes the secret s. Both proposed schemes incorporate effective, built-in techniques for share verification and the detection and identification of adversarial behaviour, directly deriving these capabilities from the fundamental evaluation and scaling properties of polynomial resultants.

The operational correctness of the polynomial construction algorithm and the main security properties of both schemes (including their threshold behaviour, privacy, and verifiability) have been demonstrated through illustrative examples and formally validated by comprehensive mathematical proofs.

The resultant-based framework established here offers new insights and tools for cryptographic design. The ability to embed verification directly via algebraic properties such as scaling is a significant advantage. This work has shown that resultants are not just tools for algebraic elimination but can be repurposed effectively for cryptographic constructions.

In summary, this chapter has laid a new foundation for constructing secret sharing schemes that are both secure and verifiable, by innovatively applying the theory of polynomial resultants. The developed schemes, algorithms, and analyses contribute to the ongoing effort to design more effective and adaptable cryptographic tools, opening promising pathways for future exploration in both classical and post-quantum cryptography.

In summary, this chapter has laid a new foundation for constructing secret sharing schemes that are both secure and verifiable, by innovatively applying the theory of polynomial resultants. The developed schemes, algorithms, and analyses contribute to the ongoing effort to design more effective and adaptable cryptographic tools, opening promising pathways for future exploration in both classical and post-quantum cryptography. Moreover, the inherent non-linearity of the resultant structure introduces a novel layer of algebraic complexity, offering a distinct alternative to the linearity of standard interpolation-based methods. Ultimately, this research validates the concept that classical elimination theory can be effectively adapted to address emerging challenges in secure distributed computing.

Chapter 8

A Diffie–Hellman–Like Key-Exchange Protocol via Commuting Skew Resultants

This chapter introduces a novel Diffie–Hellman–like key-exchange protocol that operates entirely within a non-commutative, multivariate Ore algebra. The central idea is to exploit the fact that skew resultants, defined via the Dieudonné determinant, take values in an abelianised multiplicative group; although the underlying Ore polynomials do not commute, their associated resultants (coming from matrices of the same size over a skew field) do commute. This *commutativity in the abelianised image* is cryptographically meaningful in two complementary ways.

First, it yields a simplified key-generation mechanism. Earlier schemes in multivariate Ore rings, such as the construction of Burger and Heinle [18], require participants to select private keys from carefully engineered subsets of commuting polynomials embedded in a larger non-commutative ring. The search for such commuting subsets is technically delicate, computationally expensive, and risks shrinking the effective key space to a highly structured subset that may itself become a target for algebraic attacks. By contrast, in the protocol presented

here, each party chooses private keys as generic elements subject only to natural degree and support constraints; commutativity is obtained automatically at the level of skew resultants, with no need for an explicit commuting subalgebra. This reduces key-generation complexity and leads to a conceptually cleaner and more uniform key space.

Secondly, the use of commuting skew resultants allows the security assumptions to be formulated directly in terms of the difficulty of recovering or distinguishing private Ore-polynomial data from resultant values in an abelianised group. The protocol leverages the full non-commutative structure of the underlying Ore algebra for hiding information, but the exchanged public values live in a commutative image where a Diffie–Hellman–like composition law is available. This stands in contrast to earlier non-commutative schemes, where security often depends on ad hoc commutation hypotheses or on ring properties (such as the Euclidean property) that have been shown to enable efficient greatest-common-divisor attacks [48]. In the construction developed here, commutativity is not an artefact imposed on the key space but an inherent property of the skew resultant itself, which can be analysed via Dieudonné determinants and degree bounds.

The chapter formalises this algebraic framework, defines the commuting skew resultants used as cryptographic conjugators, and then describes a complete key-exchange protocol built on these objects. A comparative discussion is provided to highlight how the use of commuting skew resultants simultaneously (1) removes the need for pre-computed commuting subsets, (2) avoids known structural weaknesses of Euclidean Ore domains, and (3) leads to a clearer and more tractable security assumption that is distinct from both classical commutative Diffie–Hellman and previous non-commutative proposals.

8.1 Introduction and Background

In response to the threat that quantum computing poses to cryptosystems based on integer factorisation and discrete logarithms, research has turned to alternative

platforms, including non-commutative algebraic structures [11, 18, 114]. The underlying motivation is to identify computational problems, such as factoring or solving polynomial equations in multivariate Ore rings, that are conjectured to be infeasible for both classical and quantum computers, thereby providing a foundation for post-quantum safe cryptographic schemes.

An early proposal by Boucher et al. [11] employed univariate skew polynomials for a key-exchange scheme. However, as Dubois and Kammerer later demonstrated [48], this protocol was vulnerable to structural attacks using greatest-common-divisor (GCD) computations, a weakness rooted in the fact that the underlying univariate Ore ring is a Euclidean domain. Once the Euclidean property is present, the non-commutative structure can be partially linearised, enabling efficient recovery of private keys. To mitigate this, Burger and Heinle [18] proposed moving to multivariate Ore polynomial rings, which are not Euclidean domains and thus are not immediately susceptible to such GCD-based attacks. Their protocol, however, requires participants to select private keys from special, pre-constructed subsets of commuting polynomials. The generation and verification of these subsets are computationally costly and introduce an additional layer of structure that must be secured: if the commuting subset is too small or too regular, it may itself become the focus of algebraic or combinatorial attacks.

The approach in this chapter revisits key exchange in Ore algebras from a different perspective, building upon the theory of skew resultants developed in Chapter 4. The resultant employed here is formulated via the Dieudonné determinant. Although the determinant is computed inside a non-commutative matrix ring, its value lies in the abelianised multiplicative group of the underlying skew field, and resultants arising from matrices of the same size therefore commute with one another. In particular, if two parties start from non-commuting bivariate Ore polynomials and form suitable Sylvester-type matrices, the corresponding skew resultants commute in the target group even though the original polynomials do not. This inherent commutativity is used to realise a Diffie–Hellman–like

operation on public data without the need to impose commutativity at the level of private keys.

From a cryptographic viewpoint, this has two key consequences:

- *Reduced key-generation complexity.* Each party may select private keys as generic polynomials (within a prescribed degree and support window) in the underlying Ore algebra. There is no requirement to search for or verify membership in a commuting subalgebra. The only structure that must be enforced is that the associated Sylvester matrices have the appropriate size and non-degeneracy to yield well-defined, non-trivial skew resultants. This significantly simplifies parameter generation compared with Burger–Heinle–type constructions [18], and helps to maintain a large, relatively unstructured key space.

- *Clearer and potentially stronger security assumptions.* Public values in the protocol consist of skew resultants that lie in an abelianised group, where a Diffie–Hellman–like composition law is well defined. The hardness assumption can therefore be formulated directly as the difficulty of recovering or distinguishing the underlying non-commutative Ore polynomials from their commuting resultant images, under constraints on degrees and evaluation structure. This differs qualitatively from both the discrete-log assumption in finite fields and the assumptions behind previous non-commutative proposals, which often rely on conjugacy or word problems in groups whose algebraic structure may admit unexpected simplifications.

This chapter formalises the use of commuting skew resultants as cryptographic conjugators and develops a concrete Diffie–Hellman–like key-exchange protocol based on them. The principal contributions are as follows:

- The use of commuting skew resultants as *cryptographic conjugators* in bivariate Ore algebras is formalised, including the precise role of the Dieudonné determinant and the abelianisation of the multiplicative group.

- A complete *key-exchange protocol* is presented, detailing the algebraic setup, key generation, and shared-secret derivation, and showing how the commuting property of resultants yields a Diffie–Hellman–like structure without pre-constructed commuting subsets.
- The *algebraic correctness*, parameter selection, and security assumptions of the protocol are analysed and compared with those of earlier systems, highlighting how commuting skew resultants influence both efficiency (through simplified key generation) and security modelling (through a well-defined hardness assumption on non-commutative-to-commutative images).

8.2 The Commuting-Resultant Protocol

8.2.1 Algebraic Preliminaries

Let $\mathcal{F}$ be a finite field and, for $i = 1, 2$, let $\sigma_i \colon \mathcal{F} \to \mathcal{F}$ be field automorphisms with associated σ_i-derivations δ_i. Consider the *bivariate Ore algebra*

$$\mathcal{S} \;=\; \big(\mathcal{F}[\theta_1; \sigma_1, \delta_1]\big)[\theta_2; \sigma_2, \delta_2],$$

which is an iterated Ore extension. As established in Chapter 4, for two skew polynomials $F, G \in \mathcal{S}[\theta_3; \sigma_3, \delta_3]$ (in a third indeterminate θ_3), their *skew resultant* $\mathrm{Res}_{\theta_3}(F, G)$ is the Dieudonné determinant of their σ-Sylvester matrix. A fundamental property of the Dieudonné determinant is that its values lie in an abelian group. Consequently, for any four such polynomials F_1, G_1, F_2, G_2, their resultants commute:

$$\mathrm{Res}_{\theta_3}(F_1, G_1)\ \mathrm{Res}_{\theta_3}(F_2, G_2) \;=\; \mathrm{Res}_{\theta_3}(F_2, G_2)\ \mathrm{Res}_{\theta_3}(F_1, G_1).$$

This inherent commutativity supports the protocol introduced below.

8.2.2 Protocol Description

8.2.2.1 Public Parameters

Participants publicly agree on the following:

1. A fixed bivariate Ore algebra $\mathcal{S} = \mathcal{F}[\theta_1; \sigma_1, \delta_1][\theta_2; \sigma_2, \delta_2]$.
2. A public, non-central skew polynomial $L \in \mathcal{S}$ that serves as the base element.
3. A security parameter $n \in \mathbb{N}$ giving a minimum total degree for the polynomials from which private keys are derived.

8.2.2.2 Private-Key Generation

Each participant (e.g. Alice) generates her private key by choosing two random pairs of polynomials, (F_{A1}, G_{A1}) and (F_{A2}, G_{A2}), in $\mathcal{S}[\theta_3; \sigma_3, \delta_3]$. The coefficients and θ_3-degree of these polynomials are chosen to ensure a non-trivial skew resultant. She then computes the private key components:

$$R_{A1} = \mathrm{Res}_{\theta_3}(F_{A1}, G_{A1}), \qquad R_{A2} = \mathrm{Res}_{\theta_3}(F_{A2}, G_{A2}),$$

both having a total degree of at least n. The other participant (e.g. Bob) proceeds analogously to obtain R_{B1} and R_{B2}. Because all are Dieudonné resultants of matrices of the same size, the four resultants $R_{A1}, R_{A2}, R_{B1}, R_{B2}$ commute pairwise.

8.2.2.3 Key-Exchange Protocol

The exchange phase consists of the following steps, illustrated in Figure 8.1.

Step 1 **Alice** computes $C_A := R_{A1} \cdot L \cdot R_{A2}$ and sends the result to Bob.

Step 2 **Bob** computes $C_B := R_{B1} \cdot L \cdot R_{B2}$ and sends the result to Alice.

Step 3 Both parties compute the shared secret key K:

- Alice computes $K_A = R_{A1} \cdot C_B \cdot R_{A2} = R_{A1} \cdot (R_{B1} \cdot L \cdot R_{B2}) \cdot R_{A2}$.
- Bob computes $K_B = R_{B1} \cdot C_A \cdot R_{B2} = R_{B1} \cdot (R_{A1} \cdot L \cdot R_{A2}) \cdot R_{B2}$.

The protocol is correct because all resultant keys R_{Ai}, R_{Bj} commute, ensuring that $K_A = R_{A1}R_{B1}LR_{B2}R_{A2} = R_{B1}R_{A1}LR_{A2}R_{B2} = K_B$. Thus, both users derive the same key K.

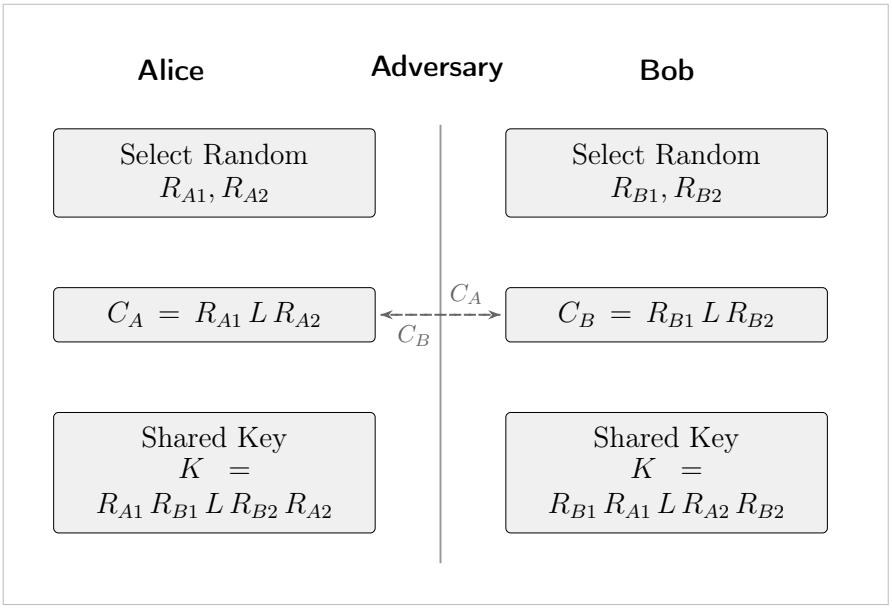

Figure 8.1: Diffie–Hellman-like key-exchange using skew resultants.

Algorithm 8 formalises the Commuting Skew-Resultant Key Exchange protocol into a unified pseudocode. This algorithm specifies the systematic procedure for parameter validation, the generation of private keys via Dieudonn'e determinants, and the secure exchange of public conjugates.

Algorithm 8: Commuting Skew-Resultant Key Exchange (Diffie–Hellman-Like over $\mathcal{S}$)

Input : Public parameters: Field $\mathcal{F}$, automorphisms σ_i and derivations δ_i (for $i = 1, 2$), and the (bivariate) Ore algebra $\mathcal{S}$
Public base element $L \in \mathcal{S}$ (non-central), security parameter $n \in \mathbb{N}$
Extension $\mathcal{S}[\theta_3; \sigma_3, \delta_3]$ and Sylvester-matrix size parameter m
Output: Shared secret K (in abelianised group)

`/* Step 1: Setup Verification */`
if L *is central in* $\mathcal{S}$ **or** L *is degenerate* **then**
  return $\perp$ `/* Invalid base element */`
end
Establish eligibility constraints such that $\mathrm{Res}_{\theta_3}(F, G)$ is non-trivial for size m
Define weak-key exclusions (e.g. prohibit graded classes susceptible to commutative reduction)

`/* Step 2: Private-Key Generation (Alice) */`
repeat
  Sample pairs $(F_{A1}, G_{A1}), (F_{A2}, G_{A2}) \in \mathcal{S}[\theta_3; \sigma_3, \delta_3]$ satisfying constraints
  $R_{A1} := \mathrm{Res}_{\theta_3}(F_{A1}, G_{A1})$
  $R_{A2} := \mathrm{Res}_{\theta_3}(F_{A2}, G_{A2})$
until $\deg(R_{A1}) \geq n$ **and** $\deg(R_{A2}) \geq n$

`/* Step 3: Private-Key Generation (Bob) */`
repeat
  Sample pairs $(F_{B1}, G_{B1}), (F_{B2}, G_{B2}) \in \mathcal{S}[\theta_3; \sigma_3, \delta_3]$ satisfying constraints
  $R_{B1} := \mathrm{Res}_{\theta_3}(F_{B1}, G_{B1})$
  $R_{B2} := \mathrm{Res}_{\theta_3}(F_{B2}, G_{B2})$
until $\deg(R_{B1}) \geq n$ **and** $\deg(R_{B2}) \geq n$

`/* Step 4: Exchange of Public Conjugates */`
Alice computes public key: $C_A := R_{A1} \cdot L \cdot R_{A2}$
Bob computes public key: $C_B := R_{B1} \cdot L \cdot R_{B2}$
Exchange C_A and C_B over authenticated channel

`/* Step 5: Shared-Secret Derivation */`
Alice computes $K_A := R_{A1} \cdot C_B \cdot R_{A2}$
Bob computes $K_B := R_{B1} \cdot C_A \cdot R_{B2}$
`/* Commutativity ensures` $K_A = K_B$ `(i.e.` $R_{A1}R_{B1} = R_{B1}R_{A1}$ `etc.) */`
$K := K_A$
return K

8.3 Analysis and Discussion

8.3.1 Security Assumptions

The security of the key-exchange protocol is founded upon the presumed computational difficulty of two related problems within the Ore algebra $\mathcal{S}$; the general problem of factoring multivariate skew polynomials and a specific problem derived from the protocol's structure, which is here termed the Skew Resultant Conjugacy Problem.

Definition 8.3.1 (Skew Resultant Conjugacy Problem (SRCP))**.** *Given the public parameters $(\mathcal{S}, L)$, along with two publicly exchanged values $C_A = R_{A1} \cdot L \cdot R_{A2}$ and $C_B = R_{B1} \cdot L \cdot R_{B2}$ (where R_{Ai}, R_{Bj} are skew resultants), the SRCP is the problem of computing the shared secret key $K = R_{A1}R_{B1}LR_{B2}R_{A2}$.*

An adversary's attempt to solve the SRCP would be to first solve a factoring problem: extracting one of the private key pairs, either (R_{A1}, R_{A2}) from C_A or (R_{B1}, R_{B2}) from C_B. While this appears similar to the challenge in the work of Burger and Heinle [18], the SRCP presents more security advantages. An adversary must solve a structured factorisation problem; they must find factors that not only solve the decomposition but also possess the specific and complex algebraic structure of being a skew resultant, rather than originating from a public structure derived from pre-computed polynomials. This task is considered no easier than factoring general multivariate Ore polynomials, for which no efficient algorithm is known. Furthermore, the inherent non-uniqueness of factorisation in such rings further complicates any attacks based on algebraic reconstruction.

The following theorem formalises the relationship between the security of the protocol and the hardness of the SRCP.

Theorem 8.3.2 (Security Reduction to the SRCP)**.** *The key-exchange protocol presented in this chapter is secure against a passive adversary if and only if the Skew Resultant Conjugacy Problem (SRCP) is computationally infeasible.*

Proof. The proof consists of two directions:

1. **(SRCP hardness → Protocol security):** Assume that the SRCP is hard to solve. A passive adversary in the key-exchange protocol has access only to the public parameters $(\mathcal{S}, L)$ and the exchanged messages (C_A, C_B). The adversary's plan is to compute the shared secret $K = R_{A1}R_{B1}LR_{B2}R_{A2}$. This is precisely the definition of the SRCP. Therefore, if the SRCP is computationally infeasible, the protocol is secure against this adversary.

2. **(Protocol security → SRCP hardness):** Assume the protocol is secure.

This means that an adversary with access to $(\mathcal{S}, L, C_A, C_B)$ cannot compute K. By definition, this implies that the SRCP is hard.

Thus, breaking the protocol is equivalent to solving the SRCP. □

8.3.2 Security Analysis and Potential Attacks

The practical security of the Commuting Skew-Resultant Key Exchange protocol relies fundamentally on the hardness of the *Skew Resultant Conjugacy Problem* (SRCP). To validate this reliance, we analyse the protocol's resistance against the primary classes of algebraic attacks that have historically compromised non-commutative cryptosystems, along with the specific parameter selections required to mitigate them.

Algebraic Reduction (Linearisation Attacks)

A standard cryptanalytic strategy is to treat the coefficients of the private key polynomials as unknowns in a system of equations derived from the public key.

- **The Attack:** An adversary symbolically expands the public equation $C_A = R_{A1} \cdot L \cdot R_{A2}$, equating the coefficients of the resulting polynomial to those of the intercepted message C_A. This generates a system of quadratic equations over $\mathcal{F}$. If the number of equations (derived from the public data) exceeds the number of monomial terms in the private keys, the system can be linearised (by replacing monomials $x_i x_j$ with independent variables y_{ij}) and solved via Gaussian elimination.

- **The Defense:** The protocol counters this by enforcing a sufficiently high total degree n. In multivariate Ore extensions, the number of monomials grows combinatorially with the degree, whereas the number of equations provided by the public key grows at a slower rate. By selecting $n \geq 25$, the system becomes severely underdetermined (where the number of unknowns

far exceeds the number of equations), rendering linearisation mathematically infeasible.

Resultant Inversion (Structural Generator Attack)

This attack exploits the fact that the private key R_{A1} is not a generic high-degree polynomial, but is generated from lower-degree "seeds" (F, G).

- **The Attack:** The number of coefficients in the seed polynomials F and G is significantly smaller than in their resultant R_{A1}. An adversary may attempt to solve for polynomials F', G', P', Q' such that $\text{Res}(F', G') \cdot L \cdot \text{Res}(P', Q') = C_A$. Solving this system is potentially easier than solving for the high-degree coefficients of R_{A1} directly, as the search space is much smaller. Additionally, adversaries may try to express C_A as the output of a structured Sylvester-like matrix and apply linear algebra or resultant-based cryptanalysis.

- **The Defense:** This is mitigated by ensuring the degrees of the seed polynomials (F, G) are high enough that the system of equations for their coefficients remains computationally intractable. Furthermore, the non-uniqueness of the resultant map implies that finding *any* pair (F', G') satisfying the equation is equivalent to solving the hard Skew Polynomial Decomposition Problem (SPDP). Conservative design must also avoid sparse support patterns or special forms of L that could admit trivial or low-rank matrix representations.

GCD-Based Structural Attacks

Previous non-commutative schemes were vulnerable to attacks exploiting the Euclidean nature of univariate polynomial rings.

- **The Attack:** In earlier schemes (e.g., Boucher et al.), the underlying ring $\mathcal{F}[x; \sigma, \delta]$ was a Euclidean domain. Adversaries could compute greatest

common divisors (GCDs) of public values (e.g., $\mathrm{GCD}(C_A, L)$) to strip away masking polynomials.

- **The Defense:** The proposed protocol operates strictly within a *bivariate* (or multivariate) Ore algebra $\mathcal{S}$. Multivariate polynomial rings are not Euclidean domains; they lack a division algorithm and, consequently, a computable Euclidean GCD. This fundamental structural difference renders GCD-based attacks inapplicable.

Commutative Reduction (Weak-Key Attacks)

If the non-commutative parameters are poorly chosen, the algebraic structure may degenerate into a commutative one.

- **The Attack:** If polynomials are homogeneous (graded) and the automorphisms σ_i preserve this grading (as seen in some Weyl algebra constructions), there may exist a homomorphism mapping the non-commutative system to a commutative polynomial ring. An adversary could solve the system in the easier commutative setting and lift the solution back.

- **The Defense:** The protocol includes a mandatory "Weak-Key Exclusion" step. This enforces the rejection of graded key classes and degenerate parameters (such as identity automorphisms). Additionally, the use of a *non-central* base element L ensures that the system cannot be trivially commuted (i.e., $R_{A1}L \neq LR_{A1}$), preserving the non-commutative hardness.

Lattice Attacks and Parameter Selection

Finally, the choice of algebraic parameters dictates the resilience against geometric attacks.

- **The Attack:** An adversary might model the key recovery as a Shortest Vector Problem (SVP) in a high-dimensional lattice derived from the public algebraic relations.

- **The Defense:** To resist lattice reduction algorithms (such as LLL), the underlying field and polynomial degrees must be sufficiently large. Experimental evidence suggests that total degrees of $n \geq 25$–50 are required for security comparable to 128-bit classical systems. Furthermore, using extension fields $\mathcal{F}_{q^k}$ with $k > 1$ increases the lattice dimension, making basis reduction exponentially harder, albeit at the cost of increased computational intensity for legitimate multiplication.

8.3.3 Challenges and Solutions

The practical implementation and security of this protocol depend on several factors:

- **Insecure Key Choices:** Certain choices of polynomials or rings might unintentionally simplify the security problem. For example, in Weyl algebras, keys constructed from graded polynomials can reduce factorisation to a commutative problem, creating a significant vulnerability [18]. Protocols must include checks to avoid such known weak key classes.

- **Algebraic Attacks:** An adversary may attempt to exploit the resultant construction directly, for instance by expressing C_A and C_B as outputs of a structured Sylvester-like matrix and then applying linear algebra, Gröbner basis techniques, or resultant-based cryptanalysis in the spirit of [131]. Poorly chosen parameters (e.g. very low total degree, sparse support patterns, or special forms of L) could result in overconstrained systems where such techniques become more effective. A conservative design must therefore avoid parameter sets that admit trivial or low-rank matrix representations.

- **Linearisation Attacks:** An adversary might attempt to linearise the problem by treating products of monomials as independent variables, thereby converting the polynomial system into a large system of linear equations.

This attack becomes feasible if the number of equations (derived from public data) exceeds the number of monomial unknowns. To thwart this, the degrees of the private polynomials must be sufficiently high to ensure the number of unknowns grows faster than the available equations.

- **Performance:** Multiplication of high-degree multivariate Ore polynomials is computationally intensive. The protocol's efficiency depends on the chosen ring structure, polynomial degrees, and implementation of the underlying arithmetic. Rings where derivations δ_i are zero are generally more efficient.

- **Parameter Selection:** Choosing secure parameters $(\mathcal{F}, \mathcal{S}, L, n)$ requires a careful balance between security and performance. Experimental evidence suggests total degrees of at least 25–50 may be required for security comparable to 128-bit classical systems [18]. Using finite fields $\mathcal{F}_{q^k}$ with $k > 1$ can also help resist lattice-based attacks, which would seek to recover private keys by solving shortest-vector problems in a high-dimensional lattice derived from the public algebraic relations.

8.3.4 Comparison with Previous Work

The protocol presented herein offers distinct advantages over earlier non-commutative key-exchange proposals, such as those by Boucher et al. and Burger and Heinle. The following are the main improvements:

- No pre-computed commuting subset is required; commutativity is provided by the resultants themselves.

- Private keys are generated from a larger space of random polynomials rather than a restricted subset, which significantly enlarges the key space and enhances security.

- The protocol's foundation in a multivariate Ore algebra (which is not a

principal ideal domain) inherently resists the GCD-based attacks described by Dubois and Kammerer.

8.3.4.1 Implementation Considerations: Performance and Parameter Selection

Beyond asymptotic hardness, practical security depends on selecting parameters that are simultaneously implementable and robust against the attack classes above.

- **Performance.** Multiplication of high-degree multivariate Ore polynomials is computationally intensive. Efficiency depends strongly on the ring structure, degree choices, and the implementation of underlying arithmetic. In practice, instances with $\delta_i = 0$ are often substantially faster than those with non-trivial derivations.

- **Parameter selection.** Choosing secure parameters $(\mathcal{F}, \mathcal{S}, L, n)$ requires balancing performance against the need to avoid overconstrained algebraic systems. Experimental evidence suggests that total degrees on the order of 25–50 may be required to target security comparable to 128-bit classical levels [18]. Using extension fields $\mathcal{F}_{q^k}$ with $k > 1$ can further enlarge coefficient spaces and complicate algebraic reconstruction attempts derived from public relations.

Implementation Feasibility Note: Key Sizes and Performance Estimates

To assess the practicality of the proposed Commuting Skew-Resultant Key Exchange, indicative estimates for key sizes and computational costs are summarised below. These estimates target a security level broadly comparable to *128-bit classical security* (e.g., AES-128). It is emphasised that these are *engineering-level*

estimates rather than formal security proofs; actual performance depends heavily on the specific Ore actions (σ_i, δ_i), sparsity constraints, and the optimisation level of the underlying arithmetic (symbolic vs. finite-field).

Throughout, computations are assumed to take place in a bivariate Ore algebra over a finite field $\mathcal{F}_q$ represented with $\approx$ 64-bit elements (for instance, a 64-bit prime field $\mathcal{F}_p$ where $p \approx 2^{64}$, or a binary extension field $\mathcal{F}_{2^{64}}$). The parameter n denotes the *minimum total degree* of the private resultant factors in (x, y). This value is chosen to ensure that algebraic linearisation attacks—which require solving for the product terms of the private coefficients—remain severely underdetermined. The illustrative value $n = 30$ is provided as a baseline for feasibility analysis rather than a definitive recommendation.

Expected Key Sizes

Unlike Elliptic Curve Cryptography (ECC), where keys are compact (e.g., $\approx$ 32 bytes at 128-bit security), multivariate algebraic schemes involve the serialisation of polynomial coefficient vectors.

- **Private Keys.** In the Diffie–Hellman-like protocol, each party typically samples *two* private resultant factors (e.g., R_{A1}, R_{A2}), each generated as a skew resultant $R_{Aj} = \mathrm{Res}_{\theta_3}(F_{Aj}, G_{Aj})$ from a private seed pair (F_{Aj}, G_{Aj}) in the chosen Ore extension. Consequently, the private material can be persisted in one of two forms:

  (i) the *seed data* $\{(F_{A1}, G_{A1}), (F_{A2}, G_{A2})\}$, which is typically more compact but requires recomputation of the resultants during the handshake; or

  (ii) the *resultant factors* $\{R_{A1}, R_{A2}\}$ themselves, which avoids recomputation at the cost of storage.

  For a *dense* bivariate polynomial of total degree n, the number of monomial

terms is:

$$M(n) \;=\; \binom{n+2}{2} \;=\; \frac{(n+1)(n+2)}{2}.$$

For $n = 30$, $M(30) = 496 \approx 5 \times 10^2$ coefficients per polynomial. With 64-bit (8-byte) coefficients, this corresponds to approximately 496×8 bytes $\approx 4\,\mathrm{kB}$ per polynomial. Therefore:

- Storing two resultant factors (R_{A1}, R_{A2}) requires roughly $2 \times 4 \approx 8\,\mathrm{kB}$ in the dense worst case.
- Storing the seed polynomials is generally more compact (as seeds are of lower degree), often reducing storage substantially (depending on the chosen seed degree profiles and sparsity constraints), though this trade-off introduces computational latency.

In practice, enforcing sparsity constraints can reduce these sizes substantially; however, as skew multiplication frequently increases term density, budgeting for the dense case provides a sensible upper bound.

- **Public Exchanges.** The exchanged public values are of the form $C_A = R_{A1} \cdot L \cdot R_{A2}$. A useful first-order estimate is that total degrees are typically additive under multiplication in generic Ore domains:

$$\deg(C_A) \;\approx\; \deg(R_{A1}) + \deg(L) + \deg(R_{A2}).$$

With $\deg(R_{A1}) \approx \deg(R_{A2}) \approx n = 30$ and a modest public base element $\deg(L) \approx 10$, the expected public degree is $\deg(C_A) \approx 70$. A dense encoding at degree $D = 70$ requires $M(70) = \binom{72}{2} = 2556$ coefficients.

$$\mathrm{Size}(C_A) \approx 2556 \times 8 \text{ bytes} \approx 20\,\mathrm{kB}.$$

For a more conservative degree budget, such as $D \approx 100$, the size involves $M(100) = 5151$ coefficients, equating to $\approx 41\,\mathrm{kB}$ per message.

Comparison: These sizes are materially larger than most standardised lattice-based KEM public keys (e.g., Kyber-768 is $\approx 1.2\,\mathrm{kB}$), but are comparable to or smaller than other multivariate quadratic (MQ) proposals. They remain well within the acceptable bandwidth limits for standard internet protocols, though they may pose challenges for constrained low-bandwidth environments (e.g., LoRaWAN).

Runtime Estimates

The computational cost is dominated by the evaluation of skew resultants (via Sylvester-type matrices and Dieudonné determinant algorithms), followed by the skew polynomial multiplications required to form C_A and the shared key K.

- **Computational Complexity.** Arithmetic in $\mathcal{S}$ is inherently more expensive than commutative polynomial arithmetic because each multiplication step may necessitate the application of automorphisms σ and derivations δ. For bivariate dense polynomials of total degree d, the number of terms is $M(d) = \Theta(d^2)$. Consequently, naive skew multiplication involves $\Theta(M(d)^2) = \Theta(d^4)$ coefficient operations, multiplied by a constant factor representing the σ/δ overhead. While asymptotically faster multiplication methods exist (e.g., Karatsuba-like recursion), implementation overheads often favour standard methods for moderate d.

  The resultant generation stage reduces to structured elimination on a Sylvester-type matrix of size $m \times m$ over $\mathcal{S}$. A broad complexity envelope is:

$$\text{Cost} \;\approx\; \widetilde{O}(m^3) \text{ operations in } \mathcal{S},$$

  where $\widetilde{O}$ hides polylogarithmic factors and optimisations such as modular evaluation–interpolation. The primary practical risk is *intermediate expression swell*, which makes runtime highly sensitive to the choice of field and modular reduction strategies.

- **Estimated Latency.** Without a specific hardware benchmark, latency is best stated as an order-of-magnitude estimate. Assuming an optimised implementation in C/C++ or Rust using 64-bit machine words and modular arithmetic:
  - *Private Key Generation (Resultant Computation):* Estimated in the range of 100 ms to 1 s, depending on the matrix size m and the complexity of σ.
  - *Public Exchange and Shared Key (Skew Multiplication):* Typically faster than the resultant step, estimated in the 50–300 ms regime. Note that if the operands become fully dense, this cost can approach that of the resultant computation.

  While these timings are significantly higher than curve-based Diffie–Hellman (which often executes in < 1 ms), they are feasible for handshake-style protocols where key establishment is infrequent relative to bulk data transfer. The primary advantage justifying this cost is the potential resilience against quantum attacks that compromise standard commutative assumptions.

8.4 Formal Security Model

To rigorously evaluate the security of the proposed protocol, a game-based framework is employed involving a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. The security of the scheme relies not only on the inability of an adversary to compute the secret key (computational hardness) but also on their inability to distinguish the true secret key from a random element in the key space (decisional hardness).

8.4.1 Hardness Assumptions

Two levels of hardness are defined for the Skew Resultant Conjugacy Problem (SRCP) within the public parameter set $\Pi = (\mathcal{S}, L, n)$.

Definition 8.4.1 (Computational SRCP (C-SRCP))**.** *Given the public parameters* Π *and two valid public messages* $C_A = R_{A1}LR_{A2}$ *and* $C_B = R_{B1}LR_{B2}$*, the* ***Computational SRCP*** *is the problem of computing the shared secret* $K = R_{A1}R_{B1}LR_{B2}R_{A2}$*. The advantage of an adversary* $\mathcal{A}$ *in solving the C-SRCP is defined as:*

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{C-SRCP}}(\lambda) \;=\; \Pr\big[\mathcal{A}(\Pi, C_A, C_B) = K\big].$$

The C-SRCP is computationally infeasible *if this advantage is negligible in the security parameter* λ*.*

While the C-SRCP prevents direct key recovery, secure key exchange requires that the key appears random to an eavesdropper. This necessitates the stronger decisional assumption.

Definition 8.4.2 (Decisional SRCP (D-SRCP))**.** *Given a tuple* (Π, C_A, C_B, Z)*, where* C_A, C_B *are valid public messages and* Z *is a candidate key, the* Decisional SRCP *is the problem of deciding whether* Z *is the actual shared secret* K *or a random element chosen uniformly from the key space* $\mathcal{K}$*. The adversary's advantage in distinguishing these cases is defined as:*

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{D-SRCP}}(\lambda) \;=\; \big|\Pr[\mathcal{A}(\Pi, C_A, C_B, K) = 1] - \Pr[\mathcal{A}(\Pi, C_A, C_B, Z_{\mathrm{rand}}) = 1]\big|,$$

where K *is the real protocol key derived from* C_A, C_B*, and* $Z_{\mathrm{rand}} \xleftarrow{\$} \mathcal{K}$*. The D-SRCP is considered* hard *if this advantage is negligible.*

8.4.2 Key Indistinguishability Game (IND-KE-PASS)

The confidentiality of the established key under a passive attack (eavesdropping) is modelled using the standard notion of *Key Indistinguishability*. This is for-

malised via the following game, $\mathsf{Game}_{\mathrm{IND-KE}}$, played between $\mathcal{C}$ and $\mathcal{A}$:

1. **Setup:** The challenger $\mathcal{C}$ runs the setup algorithm to generate parameters Π and provides them to $\mathcal{A}$.
2. **Execution:** $\mathcal{C}$ generates two pairs of private keys (R_{A1}, R_{A2}) and (R_{B1}, R_{B2}) according to the protocol specification. $\mathcal{C}$ computes the corresponding public messages C_A and C_B and sends them to $\mathcal{A}$.
3. **Challenge:**
   - $\mathcal{C}$ computes the real shared secret $K_1 = R_{A1} C_B R_{A2}$.
   - $\mathcal{C}$ selects a random element $K_0 \xleftarrow{\$} \mathcal{K}$ from the same distribution as valid keys.
   - $\mathcal{C}$ flips a fair coin $b \in \{0, 1\}$ and sends the challenge K_b to $\mathcal{A}$.
4. **Guess:** The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.

The adversary succeeds if $b' = b$. The advantage of the adversary in this game is defined as:

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{IND-KE}}(\lambda) \;=\; \left|2 \cdot \Pr[b' = b] - 1\right|.$$

Theorem 8.4.3 (Security Reduction)**.** *If the Decisional Skew Resultant Conjugacy Problem (D-SRCP) is hard, then the proposed key-exchange protocol is* secure *in the sense of Key Indistinguishability (IND-KE-PASS).*

Proof. The proof relies on a direct reduction. Suppose there exists a PPT adversary $\mathcal{A}$ with a non-negligible advantage ϵ in $\mathsf{Game}_{\mathrm{IND-KE}}$. A simulator $\mathcal{S}$ can be constructed that solves the D-SRCP with the same advantage ϵ. Upon receiving a D-SRCP instance (Π, C_A, C_B, Z), the simulator $\mathcal{S}$ embeds C_A and C_B as the public messages in the IND-KE game and sets the challenge key $K_b = Z$. If $\mathcal{A}$ correctly identifies Z as the real key, $\mathcal{S}$ outputs 1 (Real); otherwise, it outputs 0 (Random). Since the distributions in the game perfectly match the distributions in the D-SRCP definition, $\mathbf{Adv}_{\mathcal{S}}^{\mathrm{D-SRCP}}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{\mathrm{IND-KE}}(\lambda)$. Thus, if the D-SRCP is hard, $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{IND-KE}}(\lambda)$ must be negligible. □

8.5 Enhancements for Efficiency and Scalability

To improve the practical scalability and security of the scheme, the following enhancements can be considered:

- **Efficiency Optimisations:** Restrict the secret polynomials F and G to be *sparse* (having few non-zero coefficients). This significantly reduces the computational cost of the skew-resultant calculation (Res_θ) without necessarily compromising the degree size n, offering a better security-to-performance ratio.

- **Security Scalability:** Extend the protocol with a key confirmation step. After computing K, parties exchange $H(K, \mathrm{ID}_A, \mathrm{ID}_B)$, where H is a secure cryptographic hash function. This ensures that both parties have agreed upon the same key and prevents active man-in-the-middle attacks that might disturb the commutative property.

8.5.1 Post-Quantum Potential

A notable advantage of this protocol is its potential for post-quantum security. The hardness of factoring elements in certain Ore polynomial rings is conjectured to resist quantum algorithms. This is based on the observation that factors can have a bit-size that is exponentially larger than that of the original polynomial [18]. If the certificate for a problem (i.e. the factors) is not representable in a size polynomial in the input, the problem may not belong to the complexity class NP. It is widely conjectured that NP-complete problems may not have efficient quantum solutions, suggesting problems potentially outside NP would be even more difficult to solve. While this phenomenon is less studied over finite fields, the general difficulty of factoring multivariate Ore polynomials provides a strong basis for considering this approach as a candidate for post-quantum cryptography.

8.6 Chapter Summary

This chapter has presented a Diffie–Hellman-like key-exchange protocol founded upon the inherent commutativity of skew resultants in bivariate Ore algebras. This approach simplifies earlier non-commutative designs by eliminating the need for specially crafted commuting subsets, thereby enlarging the key space and avoiding known vulnerabilities. The security of the protocol depends on the computational difficulty of both factoring multivariate Ore polynomials and solving the Skew Resultant Conjugacy Problem. This work demonstrates that fundamental properties of algebraic objects can provide elegant solutions in cryptographic design, opening new pathways for research at the intersection of non-commutative algebra and public-key cryptography.

Chapter 9

Overall Conclusion and Future Outlook

This thesis has presented a comprehensive investigation into the theory, computation, and application of polynomial resultants, linking fundamental advancements in non-commutative algebra with innovative developments in cryptographic scheme design. The research detailed herein successfully addressed key challenges and gaps in both the understanding of skew (Ore) polynomial resultants and their practical utilisation, particularly in the rapidly growing field of secure information sharing.

Significant theoretical advances have been made in extending resultant theory to non-commutative domains, primarily through the generalisation of classical algebra to skew polynomial rings (Ore extensions). A key development detailed in this work is the application of the Dieudonné determinant to define resultants in these structures, moving beyond standard scalar determinants to accommodate the non-abelian nature of the underlying fields. Furthermore, the formal algebraic characterisation of the skew resultant has provided the necessary framework to detect common factors, enabling the correct evaluation and interpolation logic required for the robust non-commutative secret sharing schemes presented.

Building upon these theoretical foundations, the thesis introduced two novel

secret sharing schemes: the Dealer-Side and Participant-Side Scaling schemes. These constructions demonstrate that resultants are not merely tools for elimination but can function as cryptographic primitives that inherently support verifiability. By embedding the secret within the resultant of bivariate polynomials, the schemes achieve a unique form of structural integrity, where any changes in the shares disrupts the algebraic properties required for reconstruction. This method provides a distinct alternative to standard interpolation-based approaches, offering enhanced security features such as proactive adversary detection and potential resistance against quantum-enabled threats.

However, implementing these methods in real-world cryptographic systems presents notable challenges, primarily regarding computational efficiency and infrastructure. Arithmetic operations in skew fields are inherently more intensive than in standard finite fields, potentially leading to slower encryption and decryption speeds. Additionally, the lack of standardised, optimised software libraries for skew polynomial arithmetic increases the risk of implementation errors. Further limitations identified include the communication overhead required for verification data and the complexity of selecting parameters that balance performance with resistance to structural algebraic attacks.

Consequently, the work provides a comprehensive treatment of two primary areas: the fundamental theory of non-commutative resultants and their novel application in constructing secure cryptographic systems.

9.1 Summary of Contributions

The discussion below is organised according to the main chapters of the thesis; each subsection first summarises the core achievements of its respective chapter and then highlights their implications. A concluding section presents a structured recommendations for future research works.

Chapter 4: Resultant-Based Elimination Theory

The first technical chapter formulated the algebraic foundation. It established a formal definition of the skew resultant, explored its structural properties, and presented explicit algorithms.

- Introduced a bivariate skew resultant defined via the Dieudonné determinant and proved that it can be represented by a polynomial in $\mathcal{F}[\theta_1; \sigma_1]$, despite being computed in the skew field of fractions.
- Established two equivalent characterisations of a common factor (Bézout-type and least common left multiple) and showed that both lead to the Sylvester-style matrix whose Dieudonné determinant is the resultant.
- Developed two complementary algorithms:
  (a) a direct Sylvester-style triangularisation, and
  (b) a modular evaluation–interpolation scheme that controls intermediate expression swell.
- Proved that the resultant annihilates every common solution of the operator system, thus validating its use as an elimination tool for special-function identities.

Chapter 5: Efficient Algorithms and Modular Methods

This chapter builds upon the established theoretical foundations to address the issue of computational efficiency. It develops a modular evaluation-interpolation framework specifically for Ore rings, tackling key performance bottlenecks and practical implementation challenges. The main contributions include:

- Extending the modular framework to non-commutative rings by incorporating modern evaluation and techniques suitable for Ore polynomials.

- Analysing performance inefficiencies in symbolic computation and proposing the use of root-of-unity evaluation points as a method to accelerate the interpolation stage.

Chapter 6: Implementation and Experimental Analysis

This chapter details the practical implementation and experimental validation of the resultant-based algorithms previously discussed. It focuses on assessing their performance, scalability, and correctness through a series of computational benchmarks. The key contributions of this experimental work include:

- Demonstrating the practical efficiency and enhanced scalability of the evaluation-interpolation method compared to direct symbolic approaches.
- Verifying the correctness of the implementations by cross-validating the results of the two distinct computational methods (direct and modular) against each other.
- Investigating the practical challenges of the modular approach, such as the handling of unlucky evaluation points, and confirming effective solution strategies.

Chapter 7: Resultant-Based Secret Sharing

This chapter marks the transition from algebraic theory to cryptographic application, beginning with secret sharing. It demonstrates how resultant and its properties can be transformed into reliable security protocols with verification features. The main contributions are:

- The first application of polynomial resultants for the construction of (t, n)-threshold secret sharing schemes is introduced.
- Two distinct variants are constructed, each offering different security and efficiency trade-offs:

(a) *Dealer-Side Scaling*, which uses numerical shares to ensure low computational cost for participants.

(b) *Participant-Side Scaling*, which instead employs symbolic shares within a double-structured polynomial framework, offering stronger participant-driven verification and promising post-quantum characteristics.

- A dedicated algorithm for generating bivariate polynomials whose resultant encodes a desired secret is developed, and the correctness, verifiability, and adversary-identification properties of the resulting schemes are formally established.

Chapter 8: A Diffie–Hellman–Like Key Exchange

Finally, this chapter demonstrates that the commuting-resultant principle extends naturally beyond secret sharing to other cryptographic primitives.

- A key-exchange protocol is introduced that replaces the computationally expensive requirement for pre-computed commuting subsets (as in the Burger-Heinle scheme) with the inherent commutativity of Dieudonné resultants.
- The proposed protocol is shown to offer a larger key space and require fewer public parameters, while its security relies on the hardness of two computational problems: skew-resultant conjugacy and multivariate Ore-factorisation.

9.2 Future Work and Open Problems

This thesis's findings highlight several promising avenues for future research in algebraic computation and cryptography. Accordingly, the following topics and open problems are planned for future investigation and will be covered in upcoming papers.

Future Directions in Algebraic Computation

The first part addresses future works on the algebraic theory side.

- **Higher-Variate Resultants.** A pivotal direction for future research involves extending the current bivariate framework to general multivariate skew polynomial rings, $\mathcal{F}[x_1, \ldots, x_n; \sigma, \delta]$. This generalisation necessitates defining a non-commutative analogue of the classical resultant to eliminate multiple variables simultaneously, a task significantly complicated by the non-trivial commutation relations between variables. A promising algorithmic strategy is to adapt the modular evaluation-interpolation method recursively, thus reducing high-dimensional systems into manageable lower-variate subproblems. Successfully establishing this theory would not only advance abstract algebra but also unlock new cryptographic primitives based on the hardness of solving multivariate skew systems.

- **Subresultant Chains.** Develop a full subresultant theory for skew polynomials to obtain fraction-free algorithms for GCRD computation.

- **Integration with Gröbner Basis Algorithms.** Adapt the modular evaluation-interpolation framework developed for resultants to enhance the performance of Gröbner basis computations for Ore polynomial systems. This could provide a novel approach for managing intermediate coefficient swell.

- **Optimised Modular and Parallel Implementations.** Incorporate FFT-style interpolation, explore dedicated GPU kernels, and benchmark against established solvers.

Future Directions in Cryptography

The second part focuses on how the algebraic advances can inspire new security primitives.

- **Non-Commutative Secret Sharing.** Adapt the presented schemes to use skew resultants directly, thus utilising non-unique factorisation for enhanced security.
- **Alternative Resultant Families.** Evaluate Bézout, Dixon and Sparse resultants as building blocks for cryptographic primitives with different trade-offs between efficiency and security.
- **Extended Protocol Suite.** Apply resultant techniques to digital signatures, ElGamal-like encryption, zero-knowledge proofs, verifiable machine-learning protocols, etc.

Future Directions in Performance Assessment

Practical adoption of these methods depends on thorough performance assessment and analysis against emerging standards. Key directions for future work include:

- Performing a comprehensive benchmarking campaign once a general multivariate resultant engine is available.
- Investigating further optimisations, such as utilising block-matrix layouts in resultant computations, which may reduce memory traffic (data movement) and improve cache performance.

Concluding Remarks

Collectively, the thesis demonstrates that polynomial resultants, classically limited to commutative algebra, can now be lifted to non-commutative Ore settings and, in that framework, power both symbolic elimination and practical cryptography. The algebraic developments provide new tools for computer algebra, while the secret-sharing and key-exchange prototypes illustrate the practical security benefits of the theory. This work provides a foundation for further research at

the intersection of non-commutative algebra, high-performance symbolic computation, and post-quantum cryptographic design.

Bibliography

[1] Emil Artin. *Geometric Algebra*. Wiley Classics Library. Wiley, 2011. ISBN 9781118164549.

[2] Saugata Basu, Richard Pollack, Marie-Françoise Roy, Arjeh M. Cohen, Henri Cohen, David Eisenbud, Michael F. Singer, and Bernd Sturmfels. *Algorithms in Real Algebraic Geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer, Berlin, Heidelberg, 2006. ISBN 978-3-540-33098-1. doi: 10.1007/3-540-33099-2.

[3] Amos Beimel. Secret-Sharing Schemes: A Survey. In Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology*, pages 11–46, Berlin, Heidelberg, 2011. Springer. ISBN 978-3-642-20901-7. doi: 10.1007/978-3-642-20901-7_2.

[4] William W. Bell. *Special Functions for Scientists and Engineers*. Dover books on mathematics. Dover Publications, 2004. ISBN 9780486435213.

[5] Eric Berberich, Pavel Emeliyanenko, and Michael Sagraloff. An elimination method for solving bivariate polynomial systems: Eliminating the usual drawbacks. In *Proceedings of the Meeting on Algorithm Engineering and Experiments*, ALENEX '11, pages 35–47, Philadelphia, PA, USA, 2011. Society for Industrial and Applied Mathematics.

[6] Lev M. Berkovich and Vladimir G. Tsirulik. Differential resultants and some of their applications. *Differ. Equations*, 22:530–536, 1986.

[7] Pragati Bhale, Digambar Padulkar, and Jibi Abraham. Binary Voting Protocol Using Quantum Secret Sharing. In *2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, pages 1–5, November 2023.

[8] George R. Blakley. Safeguarding cryptographic keys. In *National Computer Conference*, volume 48, pages 313–317, 1979.

[9] Carlo Blundo, Alfredo De Santis, Amir Herzberg, Shai Kutten, Ugo Vaccaro, and Moti Yung. Perfectly secure key distribution for dynamic conferences. *Information and Computation*, 146(1):1–23, 1998.

[10] Delphine Boucher and Felix Ulmer. Linear codes using skew polynomials with automorphisms and derivations. *Designs, Codes and Cryptography*, 70(3), 405–431, 2013. doi: 10.1007/s10623-012-9704-4.

[11] Delphine Boucher, Philippe Gaborit, Willi Geiselmann, Olivier Ruatta, and Felix Ulmer. Key Exchange and Encryption Schemes Based on Non-commutative Skew Polynomials. In Nicolas Sendrier, editor, *Post-Quantum Cryptography*, pages 126–141, Berlin, Heidelberg, 2010. Springer. doi: 10.1007/978-3-642-12929-2_10.

[12] François Boulier, Daniel Lazard, François Ollivier, and Michel Petitot. Representation for the radical of a finitely generated differential ideal. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, ISSAC '95, pages 158–166, New York, NY, USA, 1995. ACM. ISBN 0-89791-699-9. doi: 10.1145/220346.220367.

[13] François Boulier, François Lemaire, Adrien Poteaux, and Marc Moreno Maza. An equivalence theorem for regular differential chains. *Journal of Symbolic Computation*, 93:34–55, 2019. ISSN 0747-7171. doi: 10.1016/j.jsc.2018.04.011.

[14] Nicolas Bourbaki. *Algebra II: Chapters 4 - 7*. Springer Science & Business Media, April 2003. ISBN 978-3-540-00706-7.

[15] Manuel Bronstein and Marko Petkovšek. An introduction to pseudo-linear algebra. *Theoretical Computer Science*, 157(1):3–33, 1996. ISSN 0304-3975.

[16] William Dale Brownawell. Bounds for the degrees in the nullstellensatz. *Annals of Mathematics*, 126(3):577–591, 1987. ISSN 0003486X.

[17] José L. Bueso, José Gómez-Torrecillas, and Alain Verschoren. *Algorithmic Methods in Non-Commutative Algebra: Applications to Quantum Groups*. Mathematical Modelling: Theory and Applications. Springer Netherlands, 2003. ISBN 9781402014024.

[18] Reinhold Burger and Albert Heinle. A New Primitive for a Diffie-Hellman-like Key Exchange Protocol Based on Multivariate Ore Polynomials. May 2015.

[19] Laurent Busé. Computational Algebraic Geometry. Lecture, November 2021.

[20] John Canny. Generalised characteristic polynomials. *J. Symb. Comput.*, 9 (3):241–250, March 1990. ISSN 0747-7171. doi: 10.1016/S0747-7171(08) 80012-0.

[21] Giuseppa Carrà Ferro. A resultant theory for the systems of two ordinary algebraic differential equations. *Appl. Algebra Engrg. Comm. Comput.*, 8 (6):539–560, 1997.

[22] Xavier Caruso and Jérémy Le Borgne. Fast multiplication for skew polynomials. In Michael A. Burr, Chee K. Yap, and Mohab Safey El Din, editors, *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2017, Kaiserslautern, Germany, July 25-28, 2017*, pages 77–84. ACM, 2017. doi: 10.1145/3087604.3087617.

[23] Arup Kumar Chattopadhyay, Sanchita Saha, Amitava Nag, and Sukumar Nandi. Secret sharing: A comprehensive survey, taxonomy and applications. *Computer Science Review*, 51:100608, February 2024. ISSN 1574-0137. doi: 10.1016/j.cosrev.2023.100608.

[24] Changbo Chen and Marc Moreno Maza. Algorithms for computing triangular decomposition of polynomial systems. *Journal of Symbolic Computation*, 47(6):610 – 642, 2012. ISSN 0747-7171. doi: 10.1016/j.jsc.2011.12.023. Advances in Mathematics Mechanization.

[25] Hefeng Chen and Chin-Chen Chang. A novel (t, n) secret sharing scheme based upon Euler's theorem. *Security and Communication Networks*, 2019 (1):2387358, 2019. doi: 10.1155/2019/2387358.

[26] Jingyu Chen, Haitao Deng, Huachang Su, Minghao Yuan, and Yongjun Ren. Lattice-Based Threshold Secret Sharing Scheme and Its Applications: A Survey. *Electronics*, 13(2):287, January 2024. ISSN 2079-9292. doi: 10.3390/electronics13020287.

[27] Xiaofeng Chen, Xinyi Huang, Jin Li, Jianfeng Ma, Wenjing Lou, and Duncan S. Wong. New Algorithms for Secure Outsourcing of Large-Scale Systems of Linear Equations. *IEEE Transactions on Information Forensics and Security*, 10(1):69–78, January 2015. ISSN 1556-6021. doi: 10.1109/TIFS.2014.2363765.

[28] Howard Cheng and George Labahn. A practical implementation of a modular algorithm for Ore polynomial matrices. In *Computer Mathematics*, pages 49–59, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[29] Howard Cheng and George Labahn. Modular computation for matrices of Ore polynomials. In *Computer Algebra 2006*, pages 43–66. WORLD SCIENTIFIC, August 2007. doi: 10.1142/9789812778857_0004.

[30] Xiaogang Cheng, Ren Guo, and Changli Zhou. Quantum Advantage of Threshold Changeable Secret Sharing Scheme. *International Journal of Theoretical Physics*, 63(5):109, April 2024. ISSN 1572-9575. doi: 10.1007/s10773-024-05645-4.

[31] Frédéric Chyzak and Bruno Salvy. Non-commutative elimination in Ore algebras proves multivariate identities. *J. Symbolic Comput*, 26(2):187–227, 1998.

[32] Paul M. Cohn. Free rings and their relations, 2nd ed. *London Math. Soc. Monograph No. 19*, 1985.

[33] Paul M. Cohn. *Free Ideal Rings and Localization in General Rings*. New Mathematical Monographs. Cambridge University Press, 2006. doi: 10.1017/CBO9780511542794.

[34] Paul M. Cohn. *Further Algebra and Applications*. SpringerLink : Bücher. Springer London, 2003. ISBN 9781447100393.

[35] Paul M. Cohn. *Skew Fields: Theory of General Division Rings*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2008. ISBN 9780521062947.

[36] Richard M. Cohn. *Difference Algebra*. Interscience tracts in pure and applied mathematics. Interscience Publishers, 1965.

[37] George E. Collins. Subresultant and reduced polynomial remainder sequences. *ACM Communications in Computer Algebra*, 14:128–142, 1967.

[38] George E. Collins. The calculation of multivariate polynomial resultants. *J. ACM*, 18:515–532, 1971.

[39] Keith Conrad. Infinite Galois theory (draft, CTNT 2020). Technical report, UCONN, 2020. URL https://ctnt-summer.math.uconn.edu/

wp-content/uploads/sites/1632/2020/06/CTNT-InfGaloisTheory.pdf.

[40] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Undergraduate Texts in Mathematics. Springer International Publishing, 2015. ISBN 9783319167213.

[41] Ronald Cramer, Ivan Damgård, and Ueli Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *EUROCRYPT 2000, LNCS*, volume 1807, pages 316–334, 2000.

[42] Wolfram Decker, Christian Eder, Viktor Levandovskyy, and Sharwan K. Tiwari. Modular techniques for noncommutative Gröbner bases. *Mathematics in Computer Science*, 14(1):19–33, Mar 2020. ISSN 1661-8289. doi: 10.1007/s11786-019-00412-9.

[43] Bartel Leendert van der Waerden. *Einführung in die algebraische Geometrie*. Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen mitbesonderer Berücksichtigung der Anwendungsgebiete. Verlag von Julius Springer, 1973. ISBN 9780387063614.

[44] Jean Dieudonné. Les déterminants sur un corps non commutatif. *Bulletin de la Société Mathématique de France*, 71:27–45, 1943.

[45] Fan Ding, Yihong Long, Peili Wu. Study on secret sharing for SM2 digital signature and its application. In *2018 14th International Conference on Computational Intelligence and Security (CIS)*, pages 205–209, 2018.

[46] Jian Ding, Pinhui Ke, Changlu Lin, and Huaxiong Wang. Bivariate polynomial-based secret sharing schemes with secure secret reconstruction. *Inf. Sci.*, 593(C):398–414, May 2022. ISSN 0020-0255. doi: 10.1016/j.ins.2022.02.005.

[47] Peter K. Draxl. *Skew Fields*. London Mathematical Society Lecture Note Series. Cambridge University Press, 1983. doi: 10.1017/CBO9780511661907.

[48] Vivien Dubois and Jean-Gabriel Kammerer. Cryptanalysis of cryptosystems based on noncommutative skew polynomials. Cryptology ePrint Archive, Paper 2010/411, 2010. URL https://eprint.iacr.org/2010/411.

[49] Pavel Emeliyanenko. *Harnessing the power of GPUs for problems in real algebraic geometry*. PhD thesis, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 01 2012.

[50] Pavel Emeliyanenko. Computing resultants on graphics processing units: Towards GPU-accelerated computer algebra. *Journal of Parallel and Distributed Computing*, 73(11):1494–1505, 2013. ISSN 0743-7315. doi: 10.1016/j.jpdc.2012.07.015.

[51] Ioannis Emiris and Victor Pan. Improved algorithms for computing determinants and resultants. *J. Complex.*, 21(1):43–71, February 2005. ISSN 0885-064X. doi: 10.1016/j.jco.2004.03.003.

[52] Aleksandra Lj. Erić. The resultant of non-commutative polynomials. *Matematicki Vesnik*, 60(231):3–8, 2008.

[53] Alberto Facchini and Martino Fassina. Factorization of elements in non-commutative rings, II. *Communications in Algebra*, 46(7):2928–2946, July 2018. ISSN 0092-7872. doi: 10.1080/00927872.2017.1404082.

[54] Shahin Fatima and Shish Ahmad. Secure and Effective Key Management Using Secret Sharing Schemes in Cloud Computing. *International Journal of e-Collaboration*, 16(1):1–15, January 2020. ISSN 1548-3673, 1548-3681. doi: 10.4018/IJeC.2020010102.

[55] Giuseppa Carra Ferro. A resultant theory for systems of linear partial differential equations. *Lie Groups Appl*, 1(1):47–55, 1994.

[56] Paul A. Fuhrmann. *A Polynomial Approach to Linear Algebra*. Universitext. Springer, New York, NY, 2012. ISBN 978-1-4614-0337-1. doi: 10.1007/978-1-4614-0338-8.

[57] Xiao-Shan Gao, Joris van der Hoeven, Chun-Ming Yuan, and Gui-Lin Zhang. Characteristic set method for differential-difference polynomial systems. *J. Symb. Comput.*, 44(9):1137–1163, September 2009. ISSN 0747-7171. doi: 10.1016/j.jsc.2008.02.010.

[58] Xiao-Shan Gao and Chun-Ming Yuan. Resolvent systems of difference polynomial ideals. In *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, ISSAC '06, pages 101–108, New York, NY, USA, 2006. ACM. ISBN 1-59593-276-3. doi: 10.1145/1145768.1145790.

[59] Xiao-Shan Gao, Chunming Yuan and Guilin Zhang. Ritt-Wu's characteristic set method for ordinary difference polynomial systems with arbitrary ordering. *Acta Mathematica Scientia*, 29(4):1063–1080, 2009. ISSN 0252-9602. doi: 10.1016/S0252-9602(09)60086-2.

[60] Xiao-Shan Gao, Yan Luo, and Chun-Ming Yuan. A characteristic set method for ordinary difference polynomial systems. *J. Symb. Comput.*, 44 (3):242–260, March 2009. ISSN 0747-7171. doi: 10.1016/j.jsc.2007.05.005.

[61] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, New York, NY, USA, 3rd edition, 2013. ISBN 1107039037.

[62] Israel M. Gelfand, Mikhail Kapranov, and Andrei Zelevinsky. *Discriminants, Resultants, and Multidimensional Determinants*. Modern Birkhäuser Classics. Birkhäuser Boston, 2008. ISBN 9780817647704.

[63] Mark Giesbrecht, Qiao-Long Huang, and Éric Schost. Sparse multiplication for skew polynomials. In *Proceedings of the 45th International Symposium on Symbolic and Algebraic Computation*, ISSAC 2020, pages 194–201, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450371001. doi: 10.1145/3373207.3404023.

[64] Mark Giesbrecht and Myung Sub Kim. Computing the Hermite form of a matrix of Ore polynomials. *Journal of Algebra*, 376:341–362, 2013. ISSN 0021-8693. doi: 10.1016/j.jalgebra.2012.11.033.

[65] Kenneth R. Goodearl and Robert B. Warfield. *An Introduction to Noncommutative Noetherian Rings*. London Mathematical Society Student Texts. Cambridge University Press, 1989.

[66] Richard Gustavson. *Elimination for Systems of Algebraic Differential Equations*. PhD thesis, City University of New York, 2017.

[67] Dirk Hachenberger. *Finite Fields: Normal bases and completely free elements*. The Springer International Series in Engineering and Computer Science. Springer US, 2012. ISBN 9781461562696.

[68] Dirk Hachenberger. Universal normal bases for the abelian closure of the field of rational numbers. *Acta Arithmetica*, 93, 01 2000. doi: 10.4064/aa-93-4-329-341.

[69] Zhijie Han, Xingbo Xie, Xiaoyu Du, Ying Du, and Xin He. Blockchain-Based Data Integrity Verification. In *2024 4th International Conference on Blockchain Technology and Information Security (ICBCTIS)*, pages 226–232, August 2024. doi: 10.1109/ICBCTIS64495.2024.00043.

[70] Lein Harn, Chingfang Hsu, and Zhe Xia. A novel threshold changeable secret sharing scheme. *Frontiers of Computer Science*, 16(1):161807, October 2021. ISSN 2095-2236. doi: 10.1007/s11704-020-0300-x.

[71] Lein Harn and Chi-Fang Hsu. Dynamic threshold secret reconstruction and its application to the threshold cryptography. *Information Processing Letters*, 115(11):851–857, 2015.

[72] Lein Harn, Zhe Xia, Chi-Fang Hsu, and Y. Liu. Secret sharing with secure secret reconstruction. *Information Sciences*, 519:1–8, 2020.

[73] Albert Heinle. Factorization, similarity and matrix normal forms over certain Ore domains. Master's thesis, University of RWTH Aachen, 2012.

[74] Hiroshi Hirai, Yuni Iwamasa, Taihei Oki, and Tasuku Soma. Algebraic combinatorial optimization on the degree of determinants of noncommutative symbolic matrices. *Mathematical Programming*, November 2024. ISSN 1436-4646. doi: 10.1007/s10107-024-02158-0.

[75] Hoon Hong. Ore subresultant coefficients in solutions. *Appl. Algebra Engrg. Comm. Comput.*, 12(5):421–428, 2001.

[76] Chingfang Hsu, Lein Harn, Shan Wu, and Lulu Ke. A New Efficient and Secure Secret Reconstruction Scheme (SSRS) with Verifiable Shares Based on a Symmetric Bivariate Polynomial. *Mobile Information Systems*, 2020 (1):1039898, 2020. ISSN 1875-905X. doi: 10.1155/2020/1039898.

[77] Evelyne Hubert. Factorization-free decomposition algorithms in differential algebra. *J. Symb. Comput.*, 29(4-5):641–662, April 2000. ISSN 0747-7171. doi: 10.1006/jsco.1999.0344.

[78] Sadegh Jamshidpour and Zahra Ahmadian. Security analysis of a dynamic threshold secret sharing scheme using linear subspace method. *Information Processing Letters*, 163:105994, November 2020. ISSN 0020-0190. doi: 10.1016/j.ipl.2020.105994.

[79] Maximilian Jaroschek. Improved polynomial remainder sequences for Ore polynomials. *Journal of Symbolic Computation*, 58:64–76, 2013. ISSN 0747-7171. doi: 10.1016/j.jsc.2013.05.012.

[80] Gabriela Jeronimo, Teresa Krick, Juan Sabia, and Martin Sombra. The computational complexity of the chow form. *Found. Comput. Math.*, 4(1): 41–117, February 2004. ISSN 1615-3375. doi: 10.1007/s10208-002-0078-2.

[81] Xiao Jia, Degan Wang, Di Nie, Xiaoyan Luo, and J. Z. Sun. A new threshold changeable secret sharing scheme based on the Chinese remainder theorem. *Information Sciences*, 473:13–30, 2019.

[82] Manuel Kauers and Carsten Schneider. A refined denominator bounding algorithm for multivariate linear difference equations. In *Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, ISSAC '11, pages 201–208, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0675-1. doi: 10.1145/1993886.1993919.

[83] Masoud Khalkhali. *Basic Noncommutative Geometry*. EMS series of lectures in mathematics. European Mathematical Society, 2013. ISBN 9783037191286.

[84] Frances Clare Kirwan. *Complex Algebraic Curves*. Cambridge University Press, February 1992. ISBN 978-0-521-42353-3.

[85] Tsit Yuen Lam and André Leroy. *Algebraic Conjugacy Classes and Skew Polynomial Rings*, pages 153–203. Springer Netherlands, Dordrecht, 1988. ISBN 978-94-009-2985-2.

[86] Tsit Yuen Lam. *A First Course in Noncommutative Rings*. Graduate Texts in Mathematics. Springer, 2001. ISBN 9780387951836.

[87] Tsit Yuen Lam and André Leroy. Vandermonde and Wronskian matrices over division rings. *Journal of Algebra*, 119(2):308–336, 1988. ISSN 0021-8693. doi: 10.1016/0021-8693(88)90063-4.

[88] Hendrik W. Lenstra. A normal basis theorem for infinite Galois extensions. *Indagationes Mathematicae (Proceedings)*, 88(2):221–228, 1985. ISSN 1385-7258. doi: 10.1016/1385-7258(85)90009-5.

[89] André Leroy. Noncommutative polynomial maps. *Journal of Algebra and Its Applications*, 11, 08 2012.

[90] Xin Li, Marc Moreno Maza, Raqeeb Rasheed, and Éric Schost. High-performance symbolic computation in a hybrid compiled-interpreted programming environment. In *2008 International Conference on Computational Sciences and Its Applications*, pages 331–341, June 2008. doi: 10.1109/ICCSA.2008.68.

[91] Ziming Li. *A Subresultant Theory for Linear Differential, Linear Difference, and Ore Polynomials, with Applications*. PhD thesis, Johannes Kepler University, 1996.

[92] Ziming Li and István Nemes. A modular algorithm for computing greatest common right divisors of Ore polynomials. In *In Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, pages 282–289. ACM, 1997.

[93] Ziming Li. A subresultant theory for Ore polynomials with applications. In *ISSAC '98*, 1998.

[94] Junhong Liu, Qinfei Long, Rong-Peng Liu, Wenjie Liu, Xin Cui, and Yunhe Hou. Privacy-Preserving Peer-to-Peer Energy Trading via Hybrid Secure Computations. *IEEE Transactions on Smart Grid*, 15(2):1951–1964, March 2024. ISSN 1949-3061. doi: 10.1109/TSG.2023.3293549.

[95] Siyu Liu, Felice Manganiello, and Frank R. Kschischang. Kötter interpolation in skew polynomial rings. *Des. Codes Cryptography*, 72(3):593–608, September 2014. ISSN 0925-1022. doi: 10.1007/s10623-012-9784-1.

[96] Yanxiao Liu, Chingnung Yang, Yichuan Wang, Lei Zhu, and Wenjiang Ji. Cheating identifiable secret sharing scheme using symmetric bivariate polynomial. *Information Sciences*, 453:21–29, July 2018. ISSN 0020-0255. doi: 10.1016/j.ins.2018.04.043.

[97] Maplesoft. *Maple 2024*. http://www.maplesoft.com/, 2024.

[98] Sai Krishna Deepak Maram, Fan Zhang, Lun Wang, Andrew Low, Yupeng Zhang, Ari Juels, and Dawn Song. CHURP: Dynamic-Committee Proactive Secret Sharing. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, pages 2369–2386, New York, NY, USA, November 2019. Association for Computing Machinery. doi: 10.1145/3319535.3363203.

[99] David Marker. *Topics In Algebra Elementary Algebraic Geometry*. University of Illinois at Chicago, 2003.

[100] Umberto Martínez-Peñas. Skew and linearized Reed-Solomon codes and maximum sum rank distance codes over any division ring. *Journal of Algebra*, 504:587–612, 2018. ISSN 0021-8693. doi: 10.1016/j.jalgebra.2018.02.005.

[101] Umberto Martínez-Peñas and Frank R. Kschischang. Evaluation and interpolation over multivariate skew polynomial rings. *Journal of Algebra*, 525: 111–139, 2019. ISSN 0021-8693. doi: 10.1016/j.jalgebra.2018.12.032.

[102] Umberto Martínez-Peñas and Frank R. Kschischang. Reliable and secure multishot network coding using linearized Reed-Solomon codes. *IEEE Transactions on Information Theory*, 65(8):4785–4803, 2019b. doi: 10.1109/TIT.2019.2912165.

[103] Marc Moreno Maza and Wei Pan. Solving bivariate polynomial systems on a GPU. *Journal of Physics: Conference Series*, 341:012022, feb 2012. doi: 10.1088/1742-6596/341/1/012022.

[104] John C. McConnell, J. C. Robson, and Lance W. Small. *Noncommutative Noetherian Rings*. Graduate studies in mathematics. American Mathematical Society, 2001. ISBN 9780821821695.

[105] Keju Meng, Fuyou Miao, Wenchao Huang, and Yan Xiong. Threshold changeable secret sharing with secure secret reconstruction. *Information Processing Letters*, 157:105928, May 2020. ISSN 0020-0190.

[106] Bhubaneswar Mishra. *Algorithmic Algebra*. Springer, New York, NY, 1993. ISBN 978-1-4612-8742-1. doi: 10.1007/978-1-4612-4344-1.

[107] Hamid Nejatollahi, Nikil Dutt, Sandip Ray, Francesco Regazzoni, Indranil Banerjee, and Rosario Cammarota. Post-Quantum Lattice-Based Cryptography Implementations: A Survey. *ACM Comput. Surv.*, 51(6):129:1–129:41, January 2019. ISSN 0360-0300. doi: 10.1145/3292548.

[108] Yuri V. Nesterenko. Estimates for the orders of zeros of functions of a certain class and applications in the theory of transcendental numbers. *Mathematics of the USSR-Izvestiya*, 11(2):239–270, apr 1977. doi: 10.1070/im1977v011n02abeh001710.

[109] Øystein Ore. Formale Theorie der linearen Differentialgleichungen. (Erster Teil). *Journal für die reine und angewandte Mathematik*, 167:221–234, 1932.

[110] Øystein Ore. Linear equations in non-commutative fields. *Annals of Mathematics*, 32:463, 1931.

[111] Øystein Ore. Theory of non-commutative polynomials. *Annals of Mathematics*, 34(3):480–508, 1933. ISSN 0003486X.

[112] Raqeeb Rasheed. Modular methods for solving nonlinear polynomial systems. Master's thesis, University of Western Ontario, London, Ontario, 2007.

[113] Raqeeb Rasheed. Resultant-based elimination for skew polynomials. In *2021 23rd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 11–18, 2021. doi: 10.1109/SYNASC54541.2021.00014.

[114] Raqeeb Rasheed, Ali Safaa Sadiq, and Omprakash Kaiwartya. Elimination Algorithms for Skew Polynomials with Applications in Cybersecurity. *Mathematics*, 12(20):3258, October 2024. ISSN 2227-7390. doi: 10.3390/math12203258.

[115] Joseph Fels Ritt. *Differential Equations from an Algebraic Standpoint*, volume 14. American Mathematical Society, New York, 1932.

[116] Joseph Fels Ritt. *Differential Algebra*. Dover Publications, Inc., New York, 1966.

[117] Joseph Fels Ritt. *Differential Algebra*, volume 33 of *A.M.S. Colloquium*. A.M.S., 1950.

[118] Sonia L. Rueda and J. Rafael Sendra. Linear complete differential resultants and the implicitization of linear DPPEs. *J. Symbolic Comput.*, 45(3):324–341, 2010.

[119] Parsa Sarosh, Shabir A. Parah, and Ghulam Mohiuddin Bhat. Utilization of secret sharing technology for secure communication: A state-of-the-art review. *Multimedia Tools and Applications*, 80(1):517–541, January 2021. ISSN 1573-7721. doi: 10.1007/s11042-020-09723-7.

[120] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11): 612–613, 1979.

[121] Ian Stewart and David Tall. *Algebraic Number Theory and Fermat's Last Theorem*. CRC Press, Boca Raton, 4th edition 2016.

[122] Lenny Taelman. Dieudonné determinants for skew polynomial rings. *Journal of Algebra and Its Applications*, 05(01):89–93, 2006. doi: 10.1142/S0219498806001600.

[123] Min Tang, Zheng Yang, and Zhenbing Zeng. Resultant elimination via

implicit equation interpolation. *Journal of Systems Science and Complexity*, 29:1411–1435, 10 2016. doi: 10.1007/s11424-016-4159-8.

[124] Elahe Vedadi, Yasaman Keshtkarjahromi, and Hulya Seferoglu. Efficient Coded Multi-Party Computation at Edge Networks. *IEEE Transactions on Information Forensics and Security*, 19:807–820, 2024. ISSN 1556-6021. doi: 10.1109/TIFS.2023.3326970.

[125] Yongjie Wang, Jia Chen, Qinghong Gong, Xuehu Yan, and Yuyuan Sun. Weighted Polynomial-Based Secret Image Sharing Scheme with Lossless Recovery. *Security and Communication Networks*, 2021(1):5597592, 2021. ISSN 1939-0122. doi: 10.1155/2021/5597592.

[126] William J. Wickless. *A First Graduate Course in Abstract Algebra*. CRC Press, Boca Raton, November 2017. doi: 10.1201/9781315273228.

[127] Wen-Tsun Wu. Basic principles of mechanical theorem proving in elementary geometries. *J. Sys. Sci. and Math. Scis*, 4:207–235, 1984.

[128] Wen-Tsun Wu. A zero structure theorem for polynomial equations solving. *MM Research Preprints*, 1:2–12, 1987.

[129] Wen-Tsun Wu. On the decision problem and the mechanization of theorem-proving in elementary geometry. *Scientia Sinica*, 21(2):159–172, 1978.

[130] Wen-tsun Wu. A constructive theory of differential algebraic geometry based on works of J.F. Ritt with particular applications to mechanical theorem-proving of differential geometries. In Chaohao Gu, Marcel Berger, and Robert L. Bryant, editors, *Differential Geometry and Differential Equations*, pages 173–189, Berlin, Heidelberg, 1987. Springer. ISBN 978-3-540-47883-6. doi: 10.1007/BFb0077689.

[131] Hong-Sen Yang, Qun-Xiong Zheng, Jing Yang, Quan-Feng Liu, and Deng Tang. A New Security Evaluation Method Based on Resultant

for Arithmetic-Oriented Algorithms. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024*, pages 457–489, Singapore, 2025. Springer Nature. ISBN 978-981-9609-41-3. doi: 10.1007/978-981-96-0941-3_15.

[132] Doron Zeilberger. A holonomic systems approach to special functions identities. *J. Comput. Appl. Math*, 32(3):321–368, 1990.

[133] Hanlin Zhang, Jia Yu, Chengliang Tian, Pu Zhao, Guobin Xu, and Jie Lin. Cloud Storage for Electronic Health Records Based on Secret Sharing With Verifiable Reconstruction Outsourcing. *IEEE Access*, 6:40713–40722, 2018. ISSN 2169-3536. doi: 10.1109/ACCESS.2018.2857205.

[134] Daniel Zwillinger. Handbook of differential equations. *Academic Press, San Diego, CA, 3rd Ed*, 1998.