UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

Dissertation

Weyl Gröbner Basis Cryptosystems

Rashid Ali

Eingereicht an der Fakultät für Informatik und Mathematik
der Universität Passau als Dissertation zur Erlangung des
Grades eines Doktors der Naturwissenschaften

Submitted to the Department of Informatics and Mathematics
of the Universität Passau in Partial Fulfilment of the Requirements
for the Degree of a Doctor in the Domain of Science

Betreuer / Advisor:
Prof. Dr. Martin Kreuzer
Universität Passau

April 2011

To

my Parents,

my wife *Samina*,

my kids, *Ahmed* and *Maheen* ...

Abstract

In this thesis, we shall consider a certain class of algebraic cryptosystems called *Gröbner Basis Cryptosystems*. In 1994, *Koblitz* introduced the *Polly Cracker* cryptosystem that is based on the theory of Gröbner basis in commutative polynomials rings. The security of this cryptosystem relies on the fact that the computation of Gröbner basis is, in general, EXPSPACE-hard. Cryptanalysis of these *commutative Polly Cracker* type cryptosystems is possible by using attacks that do not require the computation of Gröbner basis for breaking the system, for example, the attacks based on *linear algebra*. To secure these (commutative) Gröbner basis cryptosystems against various attacks, among others, *Ackermann* and *Kreuzer* introduced a general class of *Gröbner Basis Cryptosystems* that are based on the difficulty of computing module Gröbner bases over general non-commutative rings. The objective of this research is to describe a special class of such cryptosystems by introducing the *Weyl Gröbner Basis Cryptosystems*. We divide this class of cryptosystems in two parts namely the (left) Weyl Gröbner Basis Cryptosystems and Two-Sided Weyl Gröbner Basis Cryptosystems. We suggest to use Gröbner bases for left and two-sided ideals in *Weyl algebras* to construct specific instances of such cryptosystems. We analyse the resistance of these cryptosystems to the standard attacks and provide computational evidence that secure *Weyl Gröbner Basis Cryptosystems* can be built using left (resp. two-sided) Gröbner bases in Weyl algebras.

Acknowledgement

I would like to use this space to say a big 'Thank You' to many people who have helped and encouraged me through out the long and difficult process of completing my doctoral studies.

At the top of the list is the name of my supervisor, Professor Dr. *Martin Kreuzer*, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject. It has been an honour to be his Ph.D student. I appreciate all his contributions of time, ideas, and suggestions to make my Ph.D. experience productive and stimulating, and above all for providing a wonderful research and working environment in our group of 'Symbolic Computations'.

I wish to acknowledge Dr. *Viktor Levandovskyy* for helpful discussions about the computations in Weyl algebras and for providing many valuable suggestions on the topic. Funds for this work are provided by the Higher Education Commission of Pakistan and the Deutscher Akademischer Austausch Dienst (German Academic Exchange Service) . The services and support of both of these organizations are highly appreciated. At this point, I am indebted to Prof. Dr. *Graf* and Dr. *Levandovskyy* for recommending my application for the further financial support.

I am indebted to all my colleagues who have shared their expertise with me. In particular I would like to thank *S. Kasper, J. Limbek* and *S. Schuster* for their help and fruitful discussions during the development of the package `Weyl` for the computer algebra system ApCoCoA. Many thanks to my office-mate *Ehsan Ullah* for the proof-reading some parts of this work and for the fruitful discussion related to polynomial system solving.

I would also like to thank my friends *Dr Tayyab Kamran, Dr Asif Bashir, Bi-*

lal, Riaz-ur-Rehman, Izhar, Imran and *Mudassar* for being available at any time for everything and for their continuous motivation and encouragement for the completion of this work. In fact, they qualify the criteria that '*A friend in need is a friend indeed'*. My experience at Passau would not have been such a pleasurable one without the presence of all my 'new' friends living there. In particular, the presence of *Ehsan-Ullah Farman, Aamir Shahzad, Amir Chishti* and their families have made my experience of living abroad such a nice and wonderful that cannot be forgotten throughout my life.

Lastly, I would like to thank my whole family for all their love and encouragement. For my parents who raised me with love and care and supported me in all my pursuits. I wish to express my love for my lovely son *Ahmed*, and my cute daughter, *Maheen* for allowing me utilizing the time that I should have to spend with them. And most of all, I would like to express my deep gratitude to my loving, supportive, and encouraging wife *Samina* for her continuous support and patience during all the stages of this thesis. Thank you.

Rashid Ali

April 14, 2011
Universität Passau, Germany

Contents

Notations

$\mathbb{K}, \mathbb{F}_p, \mathbb{Q}$	Fields
P	a commutative polynomial ring
$x_1, \ldots, x_n$	indeterminates
$\mathbb{T}^n$	the set of terms of polynomial ring P, the K-basis of P
A_n	Weyl algebra of index n
$\partial_1, \ldots, \partial_n$	additional indeterminates for a Weyl algebra
B_n	the set of Weyl terms of A, the K-basis of A
$\sigma \tau$	term orderings
p	prime number
m	plaintext unit
c	ciphertext unit
d_c	degree of the ciphertext c
G	the secret key or the set of elements of a Gröbner basis

$\mathscr{O}_\sigma(I)$ is the complement of $\mathrm{LT}_\sigma(I)$

$\mathscr{G}$	the tuple of elements in G
H	the set of elements of a partial Gröbner basis
$\mathscr{H}$	the tuple of elements in H
Q	the public key
I, J	(left) ideals of A
I_T, J_T	two-sided ideals of A

List of Abbreviations

PKC	Public Key Cryptography
SKC	Secret Key Cryptography
PCC	Polly Cracker Cryptosystem
CAS	Computer Algebra System
CGBC	Commutative Gröbner Basis Cryptosystem
GBC	Gröbner Basis Cryptosystem
RSA	Rivest Shamir Adleman
LAA	Linear Algebra Attack
ILAA	Intelligent Linear Algebra Attack
WGBC	(left) Weyl Gröbner Basis Cryptosystem
TWGBC	Two-Sided Weyl Gröbner Basis Cryptosystem

Chapter 1

Introduction

The distance is nothing; it is only the first step that is difficult.

Anonymous

The development and study of Gröbner basis cryptosystems is an active area of research in the Gröbner basis community. It is believed that if such cryptosystems are developed successfully, they will not be threatened by the development of quantum computers. Motivated by the fact that Ackermann and Kreuzer [1] recently defined a *general* class of Gröbner basis cryptosystems, the goal of this thesis is to introduce a new *special* class of Gröbner basis cryptosystems by using ideals in Weyl algebras and to present presumably hard instances of such cryptosystems.

Why?

In 1976, the new concept of *Public Key Cryptography* presented in the historical paper *"New Directions in Cryptography"*, by *Diffie* and *Hellman* [14] has radically altered the face of modern cryptography. The security of the Diffie and Hellman protocol is based on the difficulty of computing discrete logarithms in a an abelian group. Many public-key cryptosystems have been proposed and implemented since 1976. Among them, the most prominent are the ones by *Rivest, Shamir,* and *Adleman* [44] and by *ElGamal* [17]. The security of these encryption schemes rely, respectively, on the intractability of the *integer factorization problem* (IFP) and the *discrete logarithm problem* (DLP). Furthermore, due to the improvements in algorithms for solving IFP and DLP, parameters of these cryptosystems are required to be bounded by new limits in order to achieve a reasonable level of security. For instance, 156 and 200-digit RSA numbers have already been factorized. As computers get faster, to keep using cryptology, present cryptosystems have to become stronger by using longer keys and more clever techniques. In 1999, Peter Shor [46] discovered polynomial time algorithms to solve the IFP and DLP on a 'hypothetical' quantum computer. Once quantum computers have been developed, cryptosystems based on these problems will not remain secure any more. Therefore, there is a strong need to find new encryption schemes that do not depend on these two closely related problems.

The threat of quantum computers is a very hot topic in today's world of cryptography. It has been realized that there is a great need for the development of cryptosystems which are as secure on quantum computers as on conventional computers. Multivariate cryptography is one of the main fields of research for the development of multivariate algebraic cryptosystems which are believed to be secure against attacks with quantum computers [15]. The goal of this thesis is to introduce a new algebraic multivariate public key cryptosystem based on the difficulty of computing Gröbner bases of ideals in Weyl algebras. Note that the problem of computing a Gröbner basis is totally different from the IFP and DLP. In the commutative setting, the worst case complexity of computing Gröbner bases is known to be EXPSPACE (see [36]). Before we explain how we are going to achieve our goal, let us first have a brief overview of related work.

Related Work

The question whether there exist 'secure' public-key cryptosystems based on NP-hard problems has remained open for a long time. In 1994, *Fellows* and *Koblitz* [18], introduced a new algebraic multivariate encryption scheme which became known as the Polly Cracker Cryptosystem (PCC). This encryption scheme relies on the hard problem of *polynomial system solving* over a finite field. In principle, these cryptosystems could encode NP-hard problems, but constructing a hard instance turned out to be a non-trivial matter. Koblitz's PCC works as follows: Let $K = \mathbb{F}_q$ be a finite field, where $q = p^e$ with a prime number p and $e > 0$. The encryption scheme operates in a commutative ring $P = K[x_1,\dots,x_n]$ over the field K. The public key $Q = \{p_1,\dots,p_s\}$ is set by choosing a point $(a_1,\dots,a_n) \in K^n$ such that for all $i = 1,\dots,s$, we have $p_i(a_1,\dots,a_n) = 0$. For encrypting a message $m \in K$, choose "random" polynomials $h_1,\dots,h_s \in P$ and compute the encrypted message as $c = h_1 p_1 + \cdots + h_s p_s + m$. Decryption is then achieved by evaluating c at the common-zero $(a_1,\dots,a_n)$ of p_i (see Section 3.2 for details). One can attempt to attack an instance of PCC for instance by using the following two kinds of attacks:

- *total-break attacks*, where an attacker tries to reveal the secret key or to make another equivalent secret key. In this way, the attacker will be able to decrypt successfully any encrypted message.
- *single-break attacks*, where an attacker knows the encrypted message and tries to recover the corresponding original message by using publicly available information.

The cryptanalysis of various instances of PCC have been carried out successfully. *Koblitz's* "graph perfect code instance" [25], has been broken by *Hofheins* and *Steinwandt* [23] by introducing a differential attack. *R. Steinwandt* and *M. Vasco* showed in [50] that PCC is susceptible to a chosen-ciphertext attack which is a total break attack. In [49], *Steinwandt et. al.* also describe a timing attack that may be used to reveal the secret key. The cryptosystem ENROOT [20] can be viewed as a special instance of Polly Cracker which has been successfully attacked in [6]. Here we also remark that the main weakness of PCC is that its secret key is a point $(a_1,\dots,a_n)$ in K^n and the decryption is achieved by evaluating a polynomial at this point. This fact has been exploited in most of the above mentioned attacks on an instance of a PCC.

Soon PCC was generalized (see for instance [25] and [8]) to commutative *Gröbner Basis Cryptosystem*, or CGBC for short, where the underlying hard problem of polynomial system solving was replaced by the hard problem of computing Gröbner bases of ideals in commutative polynomial rings.

In particular, for an instance of a CGBC, the secret key is a Gröbner basis $G = \{g_1,\dots,g_s\}$ of an ideal $I \subset P$ with respect to some term ordering σ. The public key is a finite subset Q of I, constructed by choosing "random" polynomials $p_1,\ldots,p_s$ of the ideal I. The messages are the polynomials that are reduced with respect to G. For sending a message m, we choose random polynomials $h_1,\ldots,h_s$ and compute the encrypted message as $c = h_1\, p_1 + \cdots + h_s\, p_s + m$. The original message m then can be recovered by reducing c with respect to the secret key G. Again, theoretically, the security of a CGBC relies on the hard problem of computing a Gröbner basis but, practically, constructing a really secure instance is a non-trivial matter.

Moreover, this generalized form of PCC is also threatened by the above mentioned two kinds of attacks. That is, there are single-break attacks like the basic linear algebra attack, the 'intelligent' linear algebra attack (see [25]), and the partial Gröbner basis attack (see [8]) as well as total break attacks like the chosen-ciphertext attack. Most of these attacks exploit the structural weaknesses of CGBC. For example, in the commutative polynomial ring setting it is very difficult to hide the terms used in the polynomials $h_1,\ldots,h_s$ for computing the encrypted message $c = h_1\, p_1 + \cdots + h_s\, p_s + m$, because in this representation, terms on the right-hand side rarely cancel. Therefore, an attacker can play with the statistics of the terms in c and in the public polynomials and can have a very high probability of success for the attacks based on linear algebra. In [8], another threat for the security of a CGBC has been raised by introducing a partial Gröbner bases attack. The success of this attack greatly depends on the successful computation of a partial Gröbner basis up to a certain degree bound. Again, in the commutative setting, this method of attack might be feasible in some cases. No computational results are provided in favour of feasibility of this attack on specific instances of CGBC, but still the way it is presented suggests that these earlier suggestions of CGBC instances met a very polemic response by the Gröbner basis community. Note that the main criticisms of this encryption scheme were single-break attacks based on linear algebra and on the computation of a partial Gröbner basis.

Later, *Ackerman* and *Kreuzer* [1] discovered that the commonly used cryptosystem, RSA, can be viewed as a special case of a general kind of Gröbner basis cryptosystem. Note that RSA has not been broken yet. It follows that, the existence of the above attacks does not mean that secure instances of Gröbner basis cryptosystems cannot be constructed at all. In fact, in the following years, several possible countermeasures against these attacks have been proposed. Moreover, several modifications, to improve the general idea, have also been investigated. For instance, *L. Ly* [34], cleverly constructed a more refined version of Polly Cracker which is known as *Polly Two* and which she believed to be secure against all these standard attacks. One instance of Polly Two has been broken recently by *R. Steinwandt* [47] using a side channel attack. Because of the proposed cryptanalysis of such commutative Gröbner basis cryptosystems, it remained an open problem to construct hard instances of such systems which are secure against all standard attacks. Another attempt can be found in [41], where *T. Rai* introduced non-commutative Polly Cracker cryptosystems. The motivation for such cryptosystems was the fact that there are ideals of non-commutative polynomial rings over finite fields that have infinite reduced Gröbner bases, and hence, theoretically, there is no chance for the usual total break attack. Moreover, by construction, the single-break attacks based on linear algebra are not possible against such cryptosystems. One major weakness here seems to be the explicit suggestion to use Gröbner bases containing only one element. Principal ideals might allow an easier recovery of the secret key from the public information through a factoring attack. Moreover, finding suitable ideals for constructing concrete instances turns out to be a difficult task.

Going further in this direction, recently, *Ackermann* and *Kreuzer* have developed the most general and intelligent technique in [1] by defining *general Gröbner Basis Cryptosystems*. This general class of cryptosystems is special in the sense that it allows well known cryptosystems, such as RSA, El-Gamal, Polly Cracker, Polly 2 and Rai's non-commutative Polly Cracker to be formulated as special cases. Although no specific instances of these cryptosystems are provided, it seems to be a promising frame-work for future cryptosystems.

In this thesis, we introduce two *special* classes of instances of General Gröbner Basis Cryptosystems by using left and two-sided Gröbner basis for ideals in Weyl algebras respectively. They will be called (left) Weyl Gröbner Basis Cryptosystems (WGBC) and Two-Sided Weyl Gröbner Basis Cryptosystems (TWGBC), respectively. They are a straightforward generalization of CGBC and can also be formulated as a special case of the very general setting used in [1].

How?

The goal of constructing left and two-sided Weyl Gröbner basis cryptosystems will be achieved by going through the following steps:

(1) Introduce Weyl Gröbner Basis Cryptosystems and present methods for key generation and implementation of the enciphering and deciphering maps.
(2) Construct hard instances of these cryptosystems.
(3) Study efficiency and security issues of these cryptosystems.

Recall that the Weyl algebra A_n of index n over a field K is the associative algebra $A_n = K\langle x_1, \ldots, x_n, \partial_1, \ldots, \partial_n \rangle$ such that $[x_i, x_j] = [\partial_i, \partial_j] = 0$ and $[\partial_i, x_j] = \delta_{ij}$, where $1 \leq i, j \leq n$ and δ_{ij} is the Kronecker delta. The computational environment of our proposed cryptosystems is some Weyl algebra A_n over a field K. For a variety of reasons, it appears necessary to use a finite base field K. The secret key, G, is a Gröbner basis of an ideal $I \subset A_n$ with respect to a term ordering σ. The message space is the K-vector space generated by a small subset $\mathscr{M}$ of $\mathscr{O}_\sigma(I)$, the complement of the set of leading terms of elements of I. The public key Q is a finite set of polynomials $p_1, \ldots, p_s$ of I. For sending a message m, we carefully choose polynomials $\ell_1, \ldots, \ell_s$ and compute the encrypted message as $c = \ell_1 p_1 + \cdots + \ell_s p_s + m$. Finally, the message m can be recovered by computing the normal remainder of c with respect to the secret key G.

Why Weyl Algebras?

With the above ingredients, we shall now explain why we feel that choosing Weyl algebras as base rings for defining a special class of general Gröbner basis cryptosystems is better than the usual CGBC setting. The reasons for choosing Weyl algebras as base rings are provided by the following facts.

(1) There is a well developed and carefully studied theory of Gröbner bases of ideals in Weyl algebras. Moreover, due to non-commutativity of A_n, the computation of Gröbner bases of ideals of A_n is usually much harder than the computation in a commutative polynomial ring P.

(2) For $n \geq 1$ the set $B_n = \{x^\alpha \partial^\beta \mid \alpha, \beta \in \mathbb{N}^n\}$ of all terms forms a K-vector space basis of A_n. Therefore, every element $f \in A_n$ has a unique *standard form* given by $f = \sum c_{\alpha,\beta} x^\alpha \partial^\beta$, where $x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$, $\partial^\beta = \partial_1^{\beta_1} \cdots \partial_n^{\beta_n}$, and $c_{\alpha,\beta} \in K \backslash \{0\}$. For example, consider the Weyl algebra $A_2 = \mathbb{F}_7[x_1, x_2, \partial_1, \partial_2]$. Then the term $\partial_1^3 \partial_2 x_1^3 x_2$ will be written in its *standard form* as $x_1^3 x_2 \partial_1^3 \partial_2 + x_1^3 \partial_1^3 + 2x_1^2 x_2 \partial_1^2 \partial_2 + 2x_1^2 \partial_1^2 - 3x_1 x_2 \partial_1 \partial_2 - 3x_1 \partial_1 - x_2 \partial_2 - 1$. This feature turns out to be helpful in performing efficient multiplication of elements of A_n.

(3) Another main reason for suggesting the use of Weyl algebras for cryptography stems from Proposition 2.1.5. This result implies that every multiplication of polynomials in Weyl algebras substantially increases the size of the support of the corresponding product. For instance, let A_2 be given as above. Then the standard form of the product of a term $x_1^2 \partial_1^3 \partial_2^2$ with another term $x_1^2 x_2^3 \partial_1$ contains 9 terms,

$$\begin{aligned} x_1^2 \partial_1^3 \partial_2^2 \cdot x_1^2 x_2^3 \partial_1 \quad = \quad & x_1^4 x_2^3 \partial_1^4 \partial_2^2 - x_1^4 x_2^2 \partial_1^4 \partial_2 - x_1^3 x_2^3 \partial_1^3 \partial_2^2 - x_1^4 x_2 \partial_1^4 + x_1^3 x_2^2 \partial_1^3 \partial_2 \\ & - x_1^2 x_2^3 \partial_1^2 \partial_2^2 + x_1^3 x_2 \partial_1^3 + x_1^2 x_2^2 \partial_1^2 \partial_2 + x_1^2 x_2 \partial_1^2. \end{aligned}$$

From this observation about the product of two terms, one can imagine what is going to happen when several polynomials containing several terms are multiplied and added together to obtain a single polynomial of A_n. This means that, when we compute the encrypted message $c = \ell_1 p_1 + \cdots + \ell_s p_s + m$ in the setting of Weyl algebras, many lower degree terms are added and the coefficients of the lower degree parts change in a way that is in general hard to predict. Later, we shall see that this phenomenon is helpful to make attacks based on linear algebra infeasible when applied to an instance of our proposed cryptosystem.

(4) In the encryption process of computing $c = \ell_1 p_1 + \cdots + \ell_s p_s + m$, the polynomials $\ell_1, \ldots, \ell_s$ can be chosen to cancel the degree forms of $\ell_j p_j$ of highest degree. By the process of converting c to its standard form, the other degree forms of $\ell_j p_j$ cancel or their coefficients are changed in c. Let us observe this effect in a simple example. Consider the Weyl algebra A_2 as given above, and let $p_1 = 2x_1 x_2^2 \partial_1 \partial_2^2 - 3x_1^2 \partial_1 + 2x_2 \partial_2 - x_1 + 1$ and $p_2 = 3x_2^3 \partial_2 + x_2^2 - x_2 \partial_2 - 3$ be the given polynomials of A_2. Choose $\ell_1 = 2x_1 x_2^2 \partial_1 \partial_2 - 3x_1 \partial_1 \partial_2 + 2x_2 \partial_2 - 3$

and $\ell_2 = x_1^2 x_2 \partial_1^2 \partial_2^2 - 2x_1^2 \partial_1^2 \partial_2 + x_1 x_2 \partial_1 \partial_2^2 + x_1 \partial_1 \partial_2^2$. Then the standard form of $c = \ell_1 p_1 + \ell_2 p_2 + 3$ is given as

$$\begin{aligned} c \;=\;& x_1^3 x_2^2 \partial_1^2 \partial_2 + 2x_1^2 x_2^2 \partial_1^2 \partial_2 + 3x_1 x_2^3 \partial_1 \partial_2^2 - x_1^2 x_2 \partial_1^2 \partial_2^2 + 2x_1^3 \partial_1^2 \partial_2 - x_1 x_2 \partial_1 \partial_2^3 \\ & -2x_1^2 x_2 \partial_1^2 + x_1^2 x_2 \partial_1 \partial_2 + x_1^2 \partial_1^2 \partial_2 - 2x_1 x_2 \partial_1 \partial_2^2 - 2x_1 x_2^2 \partial_2 + x_1 x_2 \partial_1 \partial_2 - \\ & 3x_2^2 \partial_2^2 + 2x_1 \partial_1 \partial_2^2 + 2x_1^2 \partial_1 + 2x_1 x_2 \partial_1 - 2x_1 x_2 \partial_2 - 2x_1 \partial_1 \partial_2 + 2x_1 \partial_1 + \\ & 3x_1 \partial_2 + 3x_1 \end{aligned}$$

Note here that the degrees of the polynomials p_1, p_2, ℓ_1 and ℓ_2 are 6, 4, 5, and 7, respectively, and the degree of c is not 11 but 8. This means that all terms of degree greater than 8 are cancelled. Moreover, the plaintext $m = 3$ is also not visible in c. The total number of terms in c is 21 whereas the summands $\ell_1 p_1$ and $\ell_2 p_2$ contain 22 and 19 terms respectively. That is, many terms are either cancelled or their coefficients are changed in c.

(5) All the gaps in the degrees of various homogeneous components of c can be removed, for example by including a few lower degree terms in some of the polynomials $\ell_1, \ldots, \ell_s$. In this way, the encrypted message can be made more 'random-looking'. This is a relatively difficult task in the setting of CGBC. Later, in Chapter 5, we shall see that this strategy of reducing the sparsity of the polynomial c can make the intelligent linear algebra attack harder to apply in the setting of WGBC.

(6) Our methods suggested for the key generation for an instance of a WGBC do not allow the chosen ciphertext attack to work as in the setting of CGBC. In fact, using the countermeasures suggested in [42], both WGBC and TWGBC have a built-in mechanism of recognizing 'illegal' ciphertext messages. Hence the chosen ciphertext attack fails.

(7) In contrast to the commutative setting, the computation of a partial Gröbner basis turns out to be quite hard in the Weyl algebra setting. In fact, due to the properties of Weyl multiplication, the sizes of the supports of the intermediate polynomials during the computation of partial Gröbner bases grow too large. This in turn slows down the reduction process of computing normal remainders and also increases the amount of memory required to store the intermediate results during the process of computing a Gröbner basis. Hence,

a partial Gröbner basis required for the success of the partial Gröbner basis attack is hard to compute in the setting of Weyl algebras. Several examples of left as well as two-sided ideals of A_n are given in Chapters 4 and 6 which provide the evidence that large enough partial Gröbner bases of these ideals are infeasible to compute.

(8) The setting of TWGBC turns out to be even more favourable as compared with the WGBC setting. For TWGBC, the encryption is achieved by computing the standard form of $c = \ell_1 p_1 r_1 + \cdots + \ell_s p_s r_s + m$, where m is the message to be encrypted. Now the process of multiplying p_i from the left-hand and the right-hand side by suitably choosing polynomials ℓ_i and r_i and then converting c to its standard form can really mess up the resulting encrypted message (see Section 6.2 for details). In this way, it will be very hard to predict the terms used in the polynomials for left and right multiplication with the polynomials in the public key. Moreover, this encryption scheme is not vulnerable to usual attacks based on linear algebra since it is based on two-sided ideals of Weyl algebras.

Motivated by these observations, the main part of this thesis is devoted to present a detailed study and investigation of our proposed cryptosystems.

Organization of the Thesis

This section presents an outline of the remainder of the thesis and our contribution to the field of algebraic cryptography particularly the construction of hard instances of general Gröbner basis cryptosystems as presented in [1].

In Chapter 2, we introduce Weyl algebras and give their basic properties. We emphasize that Weyl algebras in positive characteristic have properties which differ from the well-known case of characteristic 0. Then we briefly describe the fundamentals of Gröbner basis theory of left ideals in these algebras. Most of this theory is available in [24] in general setting of solvable polynomial rings and in [30] for the even more general case of G-algebras. In our case, we are mostly interested in left Gröbner bases of left ideals, and in this setting most results are similar to the corresponding results from commutative Gröbner bases theory [27], or they can be adapted from commutative Gröbner bases theory using minimal alterations. The readers familiar with the theory of Gröbner bases in commutative setting can skip this section and continue with Chapter 3. We also present an easy way of constructing non-trivial left ideals in Weyl algebras, both for positive and zero characteristic. We conclude the chapter by listing various computer algebra systems available for computations in Weyl algebras. Here we also introduce our own package `Weyl` written for the computer algebra system ApCoCoA. The details about the usage of this package have been set out in Appendix A.

Chapter 3 provides the cryptographic background with emphasis on public key cryptography. After some preliminary material on cryptography, we describe *Fellows* and *Koblitz's* [18] Polly Cracker cryptosystems and then study their cryptanalysis. In particular, we describe very serious single-break attacks based on linear algebra and a total-break attack the chosen ciphertext attack, to break an instance of Polly Cracker. Afterwards, we describe commutative Gröbner basis cryptosystems and explain a partial Gröbner bases attack on such systems. We conclude the chapter by introducing the most general class of Gröbner basis cryptosystems presented in [1].

In Chapter 4, we introduce the class of (left) Weyl Gröbner basis cryptosystems. They can be viewed as a special case of the setting used in [1]. Our main contribution is then to present methods for the key generation and implementation of these cryptosystems, such that they have resistance against the standard attacks. We constructed three explicit concrete instances of these cryptosystems which we believe to be reasonably secure.

In more detail, the security and efficiency issues of these cryptosystems are studied in Chapter 5. We provide computational evidence that our proposed cryptosystems can be built to have security against all known standard attacks. In particular, we examine the security of our concrete instances of these cryptosystems against these standard attacks. By the construction and the methods introduced in Chapter 4, we think that attacks like the chosen ciphertext attack and the partial Gröbner basis attack can be safely ignored.

Finally, Chapter 6 is devoted to introduce and study two-sided Weyl Gröbner basis cryptosystems. We briefly present the fundamentals of two-sided Gröbner basis theory following the approach in [24] and [30]. We study the structure of two-sided ideals of Weyl algebras defined over a prime field $\mathbb{F}_p$. Then we provide methods for key generation for such cryptosystems and by using these methods, we construct some concrete instances of these cryptosystems. We examine their efficiency and their security against the standard attacks. In the end, we give a brief conclusion and wrap up the chapter by presenting a decryption challenge in Section 6.6.

In Appendix A we introduce the package `Weyl` for performing various computations in Weyl algebras using the computer algebra system ApCoCoA. After a brief introduction to the package, all the functions available for performing specific computations in Weyl algebras are explained with the syntax and an example describing the usage of these functions. Appendix B contains our implementation of the basic linear algebra attack and the "intelligent" linear algebra attack, both in the commutative polynomial rings and in the setting of Weyl algebras. Finally, the last Appendix C is provided to contain the data related to various examples presented throughout the thesis.

To summarize our results, we can say that one can build hard instances of our proposed cryptosystems which have resistance against the known standard attacks proposed by cryptanalysts of Gröbner basis type cryptosystems. It seems that, in order to break a Weyl Gröbner basis cryptosystem, an attacker will have no choice except to compute a Gröbner bases of the ideal generated by the elements in the corresponding public key. In [32], the degree bound for the Gröbner bases in algebras of solvable type has been established to be doubly exponential. In general, the problem of computing Gröbner bases is EXPSPACE-hard [53]. Altogether, we believe that Weyl Gröbner basis cryptosystems have potential for further investigation. Our challenge in Section 6.6 is intended to entice the readers to get into this subject. There may be many further interesting results on computations in Weyl algebras, particularly when the base field has positive characteristic.

Some results presented in this thesis are based on the joint paper *"Weyl Gröbner Basis Cryptosystems"* [2] submitted for publication.

Chapter 2

Gröbner Bases in Weyl Algebras

In this thesis, we are going to introduce a special class of Gröbner basis cryptosystems by using Weyl algebras as the base ring. The purpose of this chapter is to introduce Weyl algebras, and their basic properties. We also introduce the computational theory of Gröbner basis for Weyl algebras and study the structure of ideals in such algebras. In fact, we describe algorithms for computing Gröbner bases of ideals in Weyl algebras. The computational complexity of these algorithms motivated us to use Weyl algebras for designing the "Weyl Gröbner Basis Cryptosystems" that we describe in chapter 4.

2.1 Weyl Algebras

In this section we shall describe the main ingredients of our proposed cryptosystem, the Weyl algebra and then present some of its basic properties that are required for establishing the theory of Gröbner basis in the Weyl algebras.

Throughout the thesis let K be a field and $n \geq 1$. The characteristic of K will be denoted by $\mathrm{char}(K)$. We define the Weyl algebra of index n as follows:

Definition 2.1.1. Let $\{x_1, \ldots, x_n, \partial_1, \ldots, \partial_n\}$ denote a set of indeterminates, and let $K\langle x_1, \ldots, x_n, \partial_1, \ldots, \partial_n\rangle$ be the free associative algebra in these indeterminates. Then the **Weyl algebra** of index n over K is the associative K-algebra

$$A_n = K\langle x_1, \ldots, x_n, \partial_1, \ldots, \partial_n\rangle / I$$

where I is the two-sided ideal generated by the following elements,

$$
\begin{aligned}
& x_i x_j - x_j x_i,\ 1 \leq i, j \leq n, \\
& \partial_i \partial_j - \partial_j \partial_i,\ 1 \leq i, j \leq n, \\
& x_i \partial_j - \partial_j x_i,\ 1 \leq i \neq j \leq n, \\
& \partial_i x_i - x_i \partial_i - 1,\ i = 1, \ldots, n
\end{aligned}
$$

The last element indicates that $\partial_i x_i \neq x_i \partial_i$ and hence A_n is not commutative. If no confusion arises, from now on we denote $(x_1, \ldots, x_n)$ and $(\partial_1, \ldots, \partial_n)$ respectively by x and ∂. The elements of A_n will be called **Weyl polynomials**.

For details on the subject, we refer to standard textbooks such as [12] in the case when the field-characteristic is zero, and to the articles [43], [52] and [7] when K has a positive characteristic. For a more general class of non-commutative Noetherian rings we refer to [37] and [19] where some properties and examples are given for Weyl algebras as a special class of solvable polynomial rings both for positive and zero characteristic of the base field.

The natural action for the Weyl Algebra A_n on a polynomial f in $K[x_1, \ldots, x_n]$ is as follows:

$$\partial_i \bullet f = \frac{\partial f}{\partial x_i}, \quad x_i \bullet f = x_i f$$

Since $K[x_1, \ldots, x_n]$ is a subring of A_n, the symbol $\bullet$ helps distinguish the above action from the product $A_n \times A_n \to A_n$. For example, if $K = \mathbb{Q}$, then $\partial_1^2 \bullet x_1^3 = 6x_1$ but $\partial_1^2 \cdot x_1^3 = x_1^3 \partial_1^2 + 6x_1^2 \partial_1 + 6x_1$. With this action of an element $\partial_i \cdot x_i \in A$ on a polynomial $f \in K[x_1, \ldots, x_n]$ and using the product rule of differentiation we immediately get the last relation, $\partial_i x_i = x_i \partial_i + 1$, of the definition 2.1.1 of A_n. In fact, we have

$$(\partial_i \cdot x_i) f = x_i \frac{\partial_i f}{\partial x_i} + f \qquad \Rightarrow \qquad \partial_i \cdot x_i = x_i \cdot \partial_i + 1$$

It is easy to describe a basis for the Weyl algebra as a K-vector space by using the multi-index notation. Let x^α and ∂^β respectively denote $x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ and $\partial_1^{\beta_1} \cdots \partial_n^{\beta_n}$. Further, for $\alpha, \beta \in \mathbb{N}^n$ with $\alpha = (\alpha_1, \ldots, \alpha_n)$ and $\beta = (\beta_1, \ldots, \beta_n)$, we write

$$|\alpha| = \alpha_1 + \cdots + \alpha_n \qquad \texttt{and} \qquad |\beta| = \beta_1 + \cdots + \beta_n$$

Definition 2.1.2. In the above notation, the elements of the form $x^\alpha \partial^\beta$ in the Weyl algebra A_n are called **(Weyl) terms**.

We denote by B_n, the set of all terms in A_n. That is, for $n \geq 1$, we let

$$B_n = \{x^\alpha \partial^\beta \mid \alpha, \beta \in \mathbb{N}^n\}$$

Proposition 2.1.3. *The elements of the set B_n, as given above, form a K-vector space basis of A_n.*

Proof. See Ch. 1 Proposition 2.1 in [12]. □

In view of Proposition 2.1.3, it is natural to write every non-zero element $f \in A_n$ as a K-linear combination of elements in the basis B_n. This way of writing elements in some unique form will be useful to perform explicit calculations with the Weyl polynomials.

Definition 2.1.4. A non-zero element f in a Weyl algebra A_n written as a K-linear combination of the elements in the K-vector space basis B_n is called an element in **standard form**.

So, every element $f \in A_n$ has a unique standard form:

$$f = \sum_{(\alpha,\beta)\in E} c_{\alpha,\beta} x^\alpha \partial^\beta \tag{2.1}$$

where $x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$, $\quad \partial^\beta = \partial_1^{\beta_1} \cdots \partial_n^{\beta_n}$, $\quad c_{\alpha,\beta} \in K\backslash\{0\}$, and where E is a finite subset of $\mathbb{N}^{2n}$.

Hence, there is a natural K-vector space isomorphism between the commutative polynomial ring in $2n$ variables $\{x_1, \ldots, x_n, \xi_1, \ldots, \xi_n\}$ and the Weyl algebra A_n. Explicitly,

$$\begin{aligned} \Psi : K[x, \xi] = K[x_1, \ldots, x_n, \xi_1, \ldots, \xi_n] &\longrightarrow A_n \\ x^\alpha \xi^\beta &\longmapsto x^\alpha \partial^\beta \end{aligned} \tag{2.2}$$

Using the defining relations of Def. 2.1.1, one can convert every element of the Weyl algebra A_n into its standard form in a straightforward way. The following result proves to be useful for writing a Weyl polynomial in its standard form and hence, can be used to perform effective multiplication of Weyl polynomials.

Proposition 2.1.5. *(a) Let $i \in \{1,\dots,n\}$, and let $k,\ell \in \mathbb{N}$. Then we have*

$$\partial_i^k x_i^\ell \;=\; \sum_{j=0}^{\min\{k,\ell\}} j!\binom{k}{j}\binom{\ell}{j}\, x_i^{\ell-j}\,\partial_i^{k-j}$$

(b) Assume that $\operatorname{char}(K)=0$, and let $t = x^\alpha\partial^\beta$ and $t' = x^{\alpha'}\partial^{\beta'}$ be two terms in A_n. Write $\alpha' = (\alpha'_1,\dots,\alpha'_n)$ and $\beta = (\beta_1,\dots,\beta_n)$. Then the representation of $t\,t'$ in the basis B_n consists of

$$\prod_{i=1}^{n}(\min\{\alpha'_i,\beta_i\}+1)$$

summands.

(c) If K is a field of positive characteristic, then the number of summands in the product of the terms t and t' of part (b) becomes

$$\textstyle\prod_{i=1}^{n}(\min\{\alpha'_i \bmod p, \beta_i \bmod p\}+1)$$

Proof. (a) We can derive the formula from the relation $\partial_i x_i = x_i\partial_i + 1$ and by induction on k. For $k=1$, we have

$$\begin{aligned}
\partial_i x_i^\ell &= (\partial_i x_i)x_i^{\ell-1}\\
&= (x_i\partial_i+1)x_i^{\ell-1}\\
&= x_i(\partial_i x_i^{\ell-1}) + x_i^{\ell-1}\\
&= x_i(x_i\partial_i+1)x_i^{\ell-2} + x_i^{\ell-1}\\
&= x_i^2\partial_i + 2x_i^{\ell-1} = \cdots = x_i^\ell\,\partial_i + \ell\, x_i^{\ell-1}\\
&= \sum_{j=0}^{1} j!\binom{1}{j}\binom{\ell}{j}\, x_i^{\ell-j}\,\partial_i^{1-j}
\end{aligned}$$

Hence the formula is true for $k=1$. We shall now prove that the formula is true for $k+1$ when it is true for k.

(1) Case ($\ell \le k$)

$$
\begin{aligned}
\partial_i^{k+1} x_i^\ell &= \partial_i^k(\partial_i x_i^\ell) \\
&= \partial_i(\partial_i^k x_i^\ell) \\
&= \partial_i(\sum_{j=0}^{\ell} j! \binom{k}{j} \binom{\ell}{j} x_i^{\ell-j}\, \partial_i^{k-j}) \\
&= \partial_i(x_i^\ell\, \partial_i^k + k\,\ell\, x_i^{\ell-1}\, \partial_i^{k-1} + \cdots + (k-\ell)\, \partial_i^{k-\ell}) \\
&= (x_i^\ell\, \partial_i + \ell\, x_i^{\ell-1})\, \partial_i^k + (k\ell x_i^{\ell-1} \partial_i + k\,\ell(\ell-1)\, x_i^{\ell-2})\, \partial_i^{k-1} + \\
&\quad \cdots + (k-l)\partial_i^{k+1-\ell} \\
&= x_i^\ell\, \partial_i^{k+1} + (k+1)\ell\, x_i^{\ell-1}\, \partial_i^k + \cdots + (k-\ell)\, \partial_i^{k+1-\ell} \\
&= \sum_{j=0}^{\ell} j! \binom{k+1}{j} \binom{\ell}{j} x_i^{\ell-j}\, \partial_i^{k+1-j}
\end{aligned}
$$

Hence the formula is true for $k+1$.

(2) Case ($\ell > k$)

$$
\begin{aligned}
\partial_i^{k+1} x_i^\ell &= \partial_i^k(\partial_i x_i^\ell) = \partial_i(\partial_i^k x_i^\ell) \\
&= \partial_i(\sum_{j=0}^{k} j! \binom{k}{j} \binom{\ell}{j} x_i^{\ell-j}\, \partial_i^{k-j}) \\
&= \partial_i(x_i^\ell\, \partial_i^k + k\,\ell\, x_i^{\ell-1}\, \partial_i^{k-1} + \frac{k(k-1)\,\ell\,(\ell-1)}{2!} x_i^{\ell-2}\, \partial_i^{k-2} \\
&\quad + \cdots + (\ell-k)\, x_i^{\ell-k}) \\
&= (x_i^\ell\, \partial_i + \ell\, x_i^{\ell-1})\, \partial_i^k + (k\,\ell\, x_i^{\ell-1} \partial_i + k\,\ell(\ell-1)\, x_i^{\ell-2})\, \partial_i^{k-1} \\
&\quad + \cdots + (\frac{(k+1)\,k\,\ell\,(\ell-1)}{2!} x_i^{\ell-2}\, \partial_i + (\ell-2)\, x_i^{\ell-3})\partial_i^{k-2} \\
&\quad (\ell-k)(x_i^{\ell-k} \partial_i + x_i^{\ell-k-1} \\
&= x_i^\ell\, \partial_i^{k+1} + (k+1)\ell\, x_i^{\ell-1}\, \partial_i^k + \frac{(k+1)k\,\ell\,(\ell-1)}{2!} x_i^{\ell-2}\, \partial_i^{k-1} \\
&\quad + \cdots + (\ell-k)\, x_i^{\ell-(k+1)} \\
&= \sum_{j=0}^{k+1} j! \binom{k+1}{j} \binom{\ell}{j} x_i^{\ell-j}\, \partial_i^{k+1-j}
\end{aligned}
$$

Again, the formula is true for $k+1$.

(b) From part (a), it follows that for each $i \in \{1, \ldots, n\}$ the number of terms in the standard form of $\partial_i^k x_i^\ell$ is $(\min\{k, \ell\} + 1)$. Hence the result follows.

(c) For $\mathrm{char}(K) > 0$ we have to replace the summation bound $\min\{k,\ell\}$in part (a) by $\min\{k \mod p,\ \ell \mod p\}$ and hence the result follows.

$\square$

We have used part (a) of this proposition to implement an algorithm for computing the product of two Weyl polynomials f and g in standard form for the computer algebra system ApCoCoA. One of the motivational factors of using Weyl polynomials for designing a secure cryptosystem is part (b) of this proposition which means that the supports are going to expand greatly with every multiplication, even if it is only the multiplication by a term. We illustrate this by the following example.

Example 2.1.6. Let $m_1 = x_1^2x_2^2x_3\partial_1^3\partial_2^4\partial_3^4$ and $m_2 = x_1^4x_2^3x_3^5\partial_1\partial_2^2\partial_3^5$ be terms of the Weyl algebra $A_3 = \mathbb{Q}\langle x_1, x_2, x_3, \partial_1, \partial_2, \partial_3\rangle$. Then the number of terms in the product $m_1\, m_2$ is $(3+1)(3+1)(4+1)$ is 80. If we replace the base field by $\mathbb{Z}_7$, then the number of terms in the product is $(\min\{4 \text{ mod } 7,\ 3 \text{ mod } 7\}+1)(\min\{3 \text{ mod } 7,\ 4 \text{ mod } 7\}+1)(\min\{4 \text{ mod } 7,\ 5 \text{ mod } 7\}+1) = (3+1)(3+1)(4+1) = 80$, whereas for the field $\mathbb{Z}_5$, this product will have $4 \cdot 4 \cdot 1 = 16$ terms.

2.2 Basic Properties

In this section, we will describe the basic properties of Weyl algebras and explain how the Weyl algebras over a field K of characteristic zero are different from the ones that are defined over a field of positive characteristic.

Definition 2.2.1. Let $t = x^\alpha\,\partial^\beta$ be a Weyl term of A_n. Then the **degree** of t is given by $\deg(t) = |\alpha| + |\beta|$.

Definition 2.2.2. Let $f = c_1t_1 + \cdots + c_st_s$ be a Weyl polynomial in standard form, where $c_i \in K \setminus \{0\}$ and $t_i \in B_n$. For $i = 1, \dots, s$, the element t_i is called a **term** of f and c_i is called the **coefficient** of f corresponding to the term t_i. The summand c_it_i in this representation of f is called a **monomial** of f. We denote by $\mathrm{Supp}(f) = \{t_1, \dots, t_s\}$, the set of all terms of f and call it the **(standard) support** of f.

Definition 2.2.3. Let $f = c_1t_1 + \cdots + c_st_s$ be a Weyl polynomial in standard form, where $c_i \in K \setminus \{0\}$ and $t_i \in B_n$. The **degree**, $\deg(f)$ of the polynomial $f \in A_n$ is then defined as

$$\deg(f) = \max\{\deg(t) \mid t \in \mathrm{Supp}(f)\}$$

Note that here $f \neq 0$ and the degree of a zero-polynomial is not defined.

Definition 2.2.4. Let $f = c_1t_1 + \cdots + c_st_s$ be a Weyl polynomial in standard form, where $c_i \in K \setminus \{0\}$ and $t_i \in B_n$. We define the **degree form**, $\mathrm{DF}(f)$, of a polynomial $f \in A_n$ to be the sum of all monomials of f having degree equal to $\deg(f)$. That is,

$$\mathrm{DF}(f) = \{\sum_j c_jt_j \mid t_j \in \mathrm{Supp}(f) \text{ and } \deg(t_j) = \deg(f)\}$$

Example 2.2.5. Consider the Weyl algebra $A_2 = \mathbb{Q}[x_1, x_2, \partial_1, \partial_2]$ and
$f = 3x_1^3x_2^2\partial_1\partial_2^2 + 7x_1^3x_2^3\partial_2^2 - 2x_2^3\partial_1^4\partial_2 - 2x_1^2\partial_1^2 + \partial_1\partial_2^2 + x_1x_2 - 2x_2 + x_1 - 5$. Then we have
$\deg(f) = 8$,
$\mathrm{DF}(f) = \{3x_1^3x_2^2\partial_1\partial_2^2 + 7x_1^3x_2^3\partial_2^2 - 2x_2^3\partial_1^4\partial_2\}$, and
$\mathrm{Supp}(f) = \{x_1^3x_2^2\partial_1\partial_2^2,\, x_1^3x_2^3\partial_2^2,\, x_2^3\partial_1^4\partial_2,\, x_1^2\partial_1^2,\, \partial_1\partial_2^2,\, x_1x_2,\, x_2,\, x_1,\, 1\}$.

Proposition 2.2.6. *For Weyl polynomials $f, g \in A_n \setminus \{0\}$, the degree satisfies following the properties:*

(1) $\deg(f+g) \leq \max\{\deg(f), \deg(g)\}$, *where* $f+g \neq 0$.

(2) $\deg(fg) = \deg(f) + \deg(g)$

(3) $\deg(fg) - \deg(gf) \leq \deg(f) + \deg(g) - 2$

Proof. see [12], Ch. 2, Theorem 1.1. □

Recall that a ring is said to be **simple** if it does not have any non-trivial two-sided ideals. For a commutative ring to be simple, it has to be a *field*. This in not true in general for non-commutative rings. In fact, for Weyl algebras we have the following proposition.

Proposition 2.2.7. *Let A_n be the Weyl algebra of index n over K. If* $\mathrm{char}(K) = 0$ *then A_n does not have any non-trivial two-sided ideals, i.e. A_n is simple.*

Proof. Consider a non-zero two-sided ideal I of A_n. Let $0 \neq f \in I$ be such that $d = \deg(f) = \min\{\deg(f') \mid f' \in I \setminus \{0\}\}$. If $d = 0$, then $f \in K$, hence $I = A_n$ and there is nothing to prove. We, therefore, assume that $d > 0$. Suppose $t = x^\alpha \partial^\beta \in \mathrm{Supp}(f)$ be such that $\deg(t) = d$ and $\beta_i \neq 0$ for some $i = 1, \ldots, n$. Since, $\partial_i x_i = x_i \partial_i + 1$, and by the supposition f has a summand $t = x^\alpha \partial^\beta$ with $\deg(t) = \deg(f) = d$ and $\beta_i \neq 0$, we have $(x_i f - f x_i) \neq 0$ (because $f x_i = x_i f + h$ with $h \neq 0$) and part (3) of Proposition 2.2.6 implies that $\deg(x_i f - f x_i) \leq d - 1$. Since I is a two-sided ideal, the element $x_i f - f x_i \in I$. This contradicts our assumption that d is minimal. Hence $\beta_i = 0$, for all i. Since $d > 0$, there exists an $i \in \{1, \ldots, n\}$ such that $\alpha_i \neq 0$. Now the element $\partial_i f - f \partial_i \neq 0$ belongs to I and has degree $d - 1$ and again we have a contradiction. Therefore the ideal $I = \{0\}$ and hence A_n is simple. □

From this proposition, one can immediately infer that every *endomorphism* of A_n is injective.

Proposition 2.2.8. *Let A_n be the Weyl algebra of index n over K. If* $\mathrm{char}(K) = 0$ *then A_n is a* ***domain****, i.e. it has no left or right zero-divisors.*

Proof. As in the case of commutative polynomial ring over a field, the proof follows from part (2) of Proposition 2.2.6. □

Proposition 2.2.9. *Let A_n be the Weyl algebra of index n over K. If K is a field of positive characteristic p then the center C_n of A_n is given by*

$$C_n = K[x_1^p, \ldots, x_n^p, \partial_1^p, \ldots, \partial_n^p]$$

It is a commutative polynomial ring in $2n$ indeterminates over K. Moreover, A_n is a free C_n-module of rank p^{2n} and an Azumaya algebra of rank p^n over C_n.

Proof. These claims are proved in [52], Lemma 3. □

In view of Propositions 2.2.7, 2.2.8, and 2.2.9, most of the time we will be using mainly left ideals in Weyl algebras over a field K of positive characteristic.

Proposition 2.2.10. *A_n is a* ***left Noetherian*** *ring. That is, every left ideal is finitely generated.*

Proof. See [12] (Ch. 8, §2). □

After giving a brief introduction to Weyl algebras and their basic properties, we are now ready to describe the Gröbner basis theory for these algebras.

2.3 Left Gröbner Bases in Weyl Algebras

In this section, we will see how one can compute Gröbner bases of ideals in Weyl algebras. In [24] a Gröbner basis theory for algebras of solvable type was introduced. Weyl algebras are special cases for these algebras (see [24], 1.9.b). Teo Mora, established in [39] a unified Gröbner basis theory for both commutative and non-commutative algebras which was further considered by H. Li in his book [33] and then by Levandovskyy in his Ph.D thesis [30]. For a computational introduction to Weyl algebras, we refer to chapter one of the book [45]. Using this approach and following the notation and terminology of the books [27] and [28], we shall now present the methods for computing Gröbner bases of ideals in Weyl algebras. The main ingredients of the theory are term orderings and the division algorithm. In this section, we define term orderings on the set B_n of all terms in the Weyl algebra A_n and then describe the left division algorithm for Weyl algebras. From now on by an **ideal** we mean a ***left ideal*** of the Weyl algebra A_n, until specified otherwise.

Definition 2.3.1. A complete ordering σ on B_n is called a **(Weyl) term ordering** if it has the following properties.

(1) An inequality $x^{\alpha}\partial^{\beta} <_{\sigma} x^{\alpha'}\partial^{\beta'}$ implies

$$x^{\alpha+\alpha''}\partial^{\beta+\beta''} <_{\sigma} x^{\alpha'+\alpha''}\partial^{\beta'+\beta''}$$

for all $\alpha, \alpha', \alpha'', \beta, \beta', \beta'' \in \mathbb{N}^n$.

(2) The ordering σ is *well-founded*, i.e. we have $1 <_{\sigma} t$ for all $t \in B_n \setminus \{1\}$.

Below we define some of the well-known *term orderings* on $B_n \subset A_n$. Basically, these are the orderings induced by corresponding well-orderings on $\mathbb{N}^{2n}$.

Definition 2.3.2. We define the **lexicographic** order (`Lex`) on the terms in B_n as follows. For two terms $t_1 = x^{\alpha}\partial^{\beta}$ and $t_2 = x^{\alpha'}\partial^{\beta'}$ in B_n we say that $t_1 >_{\mathtt{Lex}} t_2$ if and only if the left-most non-zero entry in

$$(\alpha,\beta)-(\alpha',\beta') = (\alpha_1-\alpha'_1,\dots,\alpha_n-\alpha'_n,\beta_1-\beta'_1,\dots,\beta_n-\beta'_n)$$

is positive.

Example 2.3.3. Using `Lex`, the indeterminates are ordered decreasingly, that is,

$$x_1 >_{\texttt{Lex}} x_2 >_{\texttt{Lex}} \cdots >_{\texttt{Lex}} x_n >_{\texttt{Lex}} \partial_1 >_{\texttt{Lex}} \cdots >_{\texttt{Lex}} \partial_n$$

Now consider the Weyl algebra $A_2 = K[x_1, x_2, \partial_1, \partial_2]$ and let the terms $t_1, t_2 \in B_2$ be such that $t_1 = x_1 x_2^2 \partial_2$ and $t_2 = x_2^3 \partial_1^4 \partial_2^2$. Then $t_1 >_{\texttt{Lex}} t_2$, since the difference of the exponent vectors $(\alpha, \beta) - (\alpha', \beta') = (1, -1, -4, -1)$, has a positive first non-zero component.

Definition 2.3.4. We define the **degree lexicographic** order (`DegLex`) on the terms in B_n as follows. For two terms $t_1 = x^{\alpha}\partial^{\beta}$ and $t_2 = x^{\alpha'}\partial^{\beta'}$ in B_n we say that $t_1 >_{\texttt{DegLex}} t_2$ if and only if $\deg(t_1) > \deg(t_2)$ or if $\deg(t_1) = \deg(t_2)$ and $t_1 >_{\texttt{Lex}} t_2$.

Example 2.3.5. Note that, using `DegLex` we have

$$x_1 >_{\texttt{DegLex}} x_2 >_{\texttt{DegLex}} \cdots >_{\texttt{DegLex}} x_n >_{\texttt{DegLex}} \partial_1 >_{\texttt{DegLex}} \cdots >_{\texttt{DegLex}} \partial_n$$

For example, consider the Weyl algebra $A_2 = K[x_1, x_2, \partial_1, \partial_2]$ and let the terms $t_1, t_2 \in B_2$ be as given in Example 2.3.3. Then $t_2 >_{\texttt{DegLex}} t_1$, since $\deg(t_2) = 9 > \deg(t_1) = 4$. Moreover, if $t_3 = x_1^2 \partial_2^2$ then $\deg(t_1) = \deg(t_3)$ but $t_3 >_{\texttt{Lex}} t_1$ therefore $t_3 >_{\texttt{DegLex}} t_1$.

Definition 2.3.6. For the terms in $B_n \subset A_n$ we define the **degree reverse lexicographic order** (`DegRevLex`) as follows. For two terms $t_1 = x^{\alpha}\partial^{\beta}$ and $t_2 = x^{\alpha'}\partial^{\beta'}$ in B_n we say that $t_1 >_{\texttt{DegRevLex}} t_2$ if and only if $\deg(t_1) > \deg(t_2)$ or if $\deg(t_1) = \deg(t_2)$ and the right-most non-zero entry in

$$(\alpha, \beta) - (\alpha', \beta') = (\alpha_1 - \alpha_1', \ldots, \alpha_n - \alpha_n', \beta_1 - \beta_1', \ldots, \beta_n - \beta_n')$$

is negative.

Example 2.3.7. Again we have

$$x_1 >_{\texttt{DegRevLex}} \cdots >_{\texttt{DegRevLex}} x_n >_{\texttt{DegRevLex}} \partial_1 >_{\texttt{DegRevLex}} \cdots >_{\texttt{DegRevLex}} \partial_n$$

For the terms t_1, t_2, t_3 as in Example 2.3.5, we have
$t_2 >_{\texttt{DegRevLex}} t_1$ and $t_1 >_{\texttt{DegRevLex}} t_3$ since in the difference of exponent vectors $(1, 2, 0, 1) - (2, 0, 0, 2) = (-1, 2, 0, -1)$ the right-most non-zero entry is negative.

Definition 2.3.8. A term ordering σ on B_n is called **degree compatible** if $t_1 \leq_\sigma t_2$ for $t_1, t_2 \in B_n$ implies $\deg(t_1) \leq \deg(t_2)$.

For instance, `DegLex` and `DegRevLex` are degree compatible term orderings. After fixing a term ordering σ, we now define the following.

Definition 2.3.9. Consider a non-zero Weyl polynomial $f = c_1 t_1 + \cdots + c_s t_s$ with $c_i \in K \setminus \{0\}$ and $t_i \in B_n$, where $t_1 >_\sigma \cdots >_\sigma t_s$. Then we write

$$
\begin{aligned}
\mathrm{LT}_\sigma(f) &= t_1, && \text{the } \textbf{leading term} \text{ of } f, \\
\mathrm{LC}_\sigma(f) &= c_1, && \text{the } \textbf{leading coefficient} \text{ of } f, \\
\mathrm{LM}_\sigma(f) &= c_1 t_1 && \text{the } \textbf{leading monomial} \text{ of } f.
\end{aligned}
$$

Definition 2.3.10. In the setting of Example 2.2.5, let $\sigma = \mathtt{DegRevLex}$. Then we have $\mathrm{LC}_\sigma(f) = 3$, $\mathrm{LT}_\sigma(f) = x_1^3 x_2^2 \partial_1 \partial_2^2$, and $\mathrm{LM}_\sigma(f) = 3x_1^3 x_2^2 \partial_1 \partial_2^2$.

Remark 2.3.11. For Weyl algebras, if a term ordering σ satisfies only the condition (1) of the Definition 2.3.1, then it need not be *compatible with multiplication*. That is, we do not have $\mathrm{LT}_\sigma(fg) = \mathrm{LT}_\sigma(f)\mathrm{LT}_\sigma(g)$ for all $f, g \in A_n$. For instance, let τ be a complete ordering defined by

$$x^\alpha \partial^\beta <_\tau x^{\alpha'} \partial^{\beta'} \text{ if and only if } \beta - \alpha < \beta' - \alpha' \text{ or } \beta - \alpha = \beta' - \alpha' \text{ and } \alpha > \alpha'.$$

This is not compatible with multiplication. Here we have $x\partial <_\tau 1$ and $\mathrm{LT}_\tau(\partial \cdot x\partial) = \mathrm{LT}_\tau(x\partial^2 + \partial) = \partial$. Thus in case of Weyl algebras, for a complete ordering σ on B_n to be compatible with multiplication, in addition to condition (1), it must also satisfy that $1 <_\sigma x_i \partial_i$ for all $i = 1, \ldots, n$. Hence a *well founded* ordering σ together with condition (1) automatically becomes compatible with multiplication.

Let us collect some properties of leading terms in Weyl algebras.

Proposition 2.3.12. *Let σ be a term ordering on B_n. Let $f, g \in A_n \setminus \{0\}$ be such that $\mathrm{LT}_\sigma(f) = x^\alpha \partial^\beta$ and $\mathrm{LT}_\sigma(g) = x^{\alpha'} \partial^{\beta'}$ with $\alpha, \alpha', \beta, \beta' \in \mathbb{N}^n$. Then we have*

$$\mathrm{LT}_\sigma(fg) \;=\; \mathrm{LT}_\sigma(gf) \;=\; x^{\alpha+\alpha'} \partial^{\beta+\beta'}$$

Proof. First note that from Proposition 2.1.5 it follows that for any Weyl polynomials $f\,g \in A_n \setminus \{0\}$, we have $f\,g = f \cdot g + h$ with $h <_\sigma f \cdot g$ and the polynomial $h \in A_n$

is uniquely determined from f and g. Here '$\cdot$' means the commutative multiplication of the polynomials f and g, that is assuming that all the indeterminates of A_n are commuting. Now

$$\mathrm{LT}_\sigma(fg) \;=\; \mathrm{LT}_\sigma(f\cdot g) \;=\; x^{\alpha+\alpha'}\partial^{\beta+\beta'}$$

and similarly,

$$\mathrm{LT}_\sigma(gf) \;=\; \mathrm{LT}_\sigma(g\cdot f) \;=\; \mathrm{LT}_\sigma(f\cdot g) \;=\; x^{\alpha+\alpha'}\partial^{\beta+\beta'}$$

This completes the proof. □

Definition 2.3.13. For two terms $t = x^\alpha\partial^\beta$ and $t' = x^{\alpha'}\partial^{\beta'}$ in B_n we say that t **pseudo-divides** t' if $\alpha_i \le \alpha_i'$ and $\beta_i \le \beta_i'$ for all $i = 1,\dots,n$.

Definition 2.3.14. Let $t = x^\alpha\partial^\beta$ and $t' = x^{\alpha'}\partial^{\beta'}$ be two terms in B_n. For each $i \in \{1,\dots,s\}$, let $\mu_i = \max(\alpha_i,\alpha_i'),\ \nu_i = \max(\beta_i,\beta_i')$ and $(\mu,\nu) = (\mu_1,\dots,\mu_n,\nu_1,\dots,\nu_n)$. We define the **pseudo-lcm** of t_1 and t_2 as $\mathrm{lcm}(t_1,t_2) = x^\mu\,\partial^\nu$.

Definition 2.3.15. Let σ be a term ordering on A_n and consider a left ideal $I \subset A_n$. Let G be a finite subset of I. The set G is called a **left σ-Gröbner basis** of I if and only if for any $f \in I\setminus\{0\}$ there exists $g \in G$ such that $\mathrm{LT}_\sigma(g)$ pseudo-divides $\mathrm{LT}_\sigma(f)$.

Definition 2.3.16. Let F be a subset of the Weyl algebra A_n. The **span of leading terms** of F is defined to be the K-vector subspace spanned by the set $\{\mathrm{LT}_\sigma(f)\,|\,f \in F\} \subseteq B_n$. We denote it by $\langle\mathrm{LT}_\sigma(F)\rangle_K \;=\; \langle\{LT_\sigma(f)\,|\,f\in F\}\rangle_K \subseteq A_n$.

Remark 2.3.17. Here we should remark that the standard definition of Gröbner bases via leading term ideals in commutative settings cannot be transferred directly to the case of Weyl algebras. For example consider the Weyl algebra $A_1 = K[x,\partial]$, and the set $F = \{x\partial+1, x\}$. Let I be the ideal generated by F. Then I is a proper left ideal of A_1 with reduced Gröbner basis $G = \{x\}$ and $I = \langle x\rangle$. The K-vector space $\langle\mathrm{LT}_\sigma(F)\rangle_K = \langle\mathrm{LT}_\sigma(f)\,|\,f\in I\rangle_K$ is equal to the vector space $\langle x\rangle$, whereas, the ideal generated by the set $\mathrm{LT}_\sigma(F) = \langle\{x\partial, x\}\rangle = \langle 1\rangle = A$.

However, we have a well established theory of Gröbner bases of ideals in some general non-commutative rings where Weyl algebras can be considered as special cases. For instance see [24], [30], [33], and [39] . A computational introduction to the theory of Gröbner bases of ideals in Weyl algebras is also sketched in [45].

In particular, for Weyl polynomials, there exist natural definitions of S-polynomials and an analogue of the Buchberger algorithm for computing left σ-Gröbner bases of ideals in Weyl algebras.

We are now ready to give *left division algorithm* for Weyl algebras. Just like division of polynomials in commutative polynomial rings, we can divide the standard form of a Weyl polynomial $f \in A$ by a tuple $\mathscr{G} = (g_1, \ldots, g_s)$ of Weyl polynomials in standard form. With this division, we get a representation $f = q_1 g_1 + \cdots + q_s g_s + r$ with $r, q_1, \ldots, q_s \in A_n$. The polynomial $r \in A_n$ has certain extra properties and is called the *normal remainder* of the polynomial f with respect to the tuple $\mathscr{G}$. This representation and hence the normal remainder r depends not only on the term ordering σ on B_n but also on the order of the elements in the tuple $(g_1, \ldots, g_s)$. The procedure of getting this representation is known as *left division algorithm* which is the main ingredient of the Buchberger's Algorithm 2.3.24. We now present the left division algorithm for Weyl algebras in pseudo-code.

Algorithm 2.3.18. *The Left Division Algorithm*

`Input`: $f, g_1, \ldots, g_s \in A_n \setminus \{0\}$, with $\mathscr{G} = (g_1, \ldots, g_s) \subset A_n$

`Output`: The tuple $(q_1, \ldots, q_s) \in A_n^s$ and a Weyl polynomial $r \in A_n$ such that

$$f = q_1 g_1 + \cdots + q_s g_s + r$$

$q_1 := 0. \ldots q_s := 0$, $r := 0$, and $f' := f$
while ($f' \neq 0$) **do**
  while ($\exists$ smallest $i \in \{1, \ldots, s\}$ such that
    $\mathrm{LT}_\sigma(f')$ is pseudo-divisible by $\mathrm{LT}_\sigma(g_i)$) **do**
      $q_i := q_i + \frac{\mathrm{LM}_\sigma(f')}{\mathrm{LM}_\sigma(g_i)}$
      $f' := f' - \frac{\mathrm{LM}_\sigma(f')}{\mathrm{LM}_\sigma(g_i)} \cdot g_i$
  end while
  $r := r + \mathrm{LM}_\sigma(f')$
  $f' := f' - \mathrm{LM}_\sigma(f')$
end while
return $(q_1, \ldots, q_s, r)$

Proposition 2.3.19. *The Algorithm 2.3.18 terminates and returns polynomials* $q_1,\dots,q_s$ *and* $r \in A_n$ *such that*

$$f = q_1\, g_1 + \cdots + q_s\, g_s + r$$

and such that the following conditions are satisfied

(a) Either $r = 0$ *or no element of* $\operatorname{Supp}(r)$ *is pseudo-divisible by any of the element in the set* $\{\mathrm{LT}_\sigma(g_1),\dots,\mathrm{LT}_\sigma(g_s)\}$

(b) For each $i \in \{1,\dots,s\}$*, if* $q_i \neq 0$ *then we have* $\mathrm{LT}_\sigma(q_i\, g_i) \leq_\sigma \mathrm{LT}_\sigma(f)$*.*

(c) For all $i \in \{1,\dots,s\}$*, we have* $q_i \cdot \mathrm{LT}_\sigma(g_i) \notin \langle \mathrm{LT}_\sigma(g_1),\dots,\mathrm{LT}_\sigma(g_{i-1})\rangle$*.*

The polynomials $r, q_1, \dots, q_s$ *satisfying above conditions are uniquely determined by the tuple* $\mathscr{G}$ *and the polynomial* $f \in A_n$*.*

Proof. First we note that the equation

$$f = q_1\, g_1 + \cdots + q_s\, g_s + f' + r$$

holds at each point in the Algorithm 2.3.18. This is clearly true for the starting values of $q_1,\dots,q_s$, f' and r. To show that the equation holds at each step after initializing, we note that one of two things can happen. If the next step is from the inner **while**-loop, that is, some $\mathrm{LT}_\sigma(g_i)$ divides $\mathrm{LT}_\sigma(f')$, then the lines 5) and 6) in the loop ensure from the equality

$$q_i\, g_i + f' = (q_i + \frac{\mathrm{LM}_\sigma(f')}{\mathrm{LM}_\sigma(g_i)})g_i + (f' - \frac{\mathrm{LM}_\sigma(f')}{\mathrm{LM}_\sigma(g_i)} \cdot g_i)$$

that $q_i\, g_i + f'$ remains unchanged and hence the above equation holds in this case. On the other hand, if the next step is outside this loop, then again from the lines 8) and 9) of the main **while**-loop, we see that although r and f' are changed but their sum $r + f'$ is unaltered because we have

$$r + f' = (r + \mathrm{LM}_\sigma(f')) + (f' - \mathrm{LM}_\sigma(f'))$$

Thus in any case our claim remains true.

Next, we claim that the algorithm eventually terminates. To prove the claim, note that at the jth step of the second **while**-loop, we are replacing f'_j by $f'_{j-1} -$

$\frac{\mathrm{LM}_\sigma(f'_{j-1})}{\mathrm{LM}_\sigma(g_i)} \cdot g_i$. Since $\mathrm{LT}_\sigma(f'_j) < \mathrm{LT}_\sigma(f'_{j-1})$, we obtain a set $\{\mathrm{LT}_\sigma(f'_j)\}$ of leading terms of f'_j, where for all j we have $\mathrm{LT}_\sigma(f'_j) < \mathrm{LT}_\sigma(f'_{j-1})$. Since σ is well founded, this set has a minimum and hence the inner **while**-loop terminates. Similarly at line (9) f' is replaced by $f' - \mathrm{LM}(f')$ at each step of the outer **while**-loop and hence f' becomes 0 after finite number of steps of outer **while**-loop. Therefore termination of the algorithm follows and after termination we have

$$f = q_1 g_1 + \cdots + q_s g_s + r$$

and the polynomial r in the above representation will satisfy the property (a), since each time the line 8) is executed, we are adding $\mathrm{LM}_\sigma(f')$ to r only when there does not exist an $i \in \{1, \ldots, s\}$ such that $\mathrm{LT}_\sigma(f')$ is a multiple of $\mathrm{LT}_\sigma(g_i)$.

Further, note that each time the line 5) is executed and the old and new q_i are not zero, we always have the inequality

$$\mathrm{LT}_\sigma\left((q_i + \frac{\mathrm{LM}_\sigma(f')}{\mathrm{LM}_\sigma(g_i)}) \cdot g_i\right) \leq_\sigma \max\left\{\mathrm{LT}_\sigma(q_i g_i), \mathrm{LT}_\sigma(f')\right\} \leq_\sigma \mathrm{LT}_\sigma(f)$$

The same is trivially true if the old value of q_i was zero. Thus, throughout the algorithm, property (b) holds.

Now we prove property (c). For $i \in \{1, \ldots, s\}$, note that at line 3) of the algorithm, the index i is chosen minimally. Therefore, property (c) follows from the fact that $\mathrm{LT}_\sigma(f') \notin \langle \mathrm{LT}_\sigma(g_1), \ldots, \mathrm{LT}_\sigma(g_{i-1}) \rangle$, where $\mathrm{LT}_\sigma(f') = \frac{1}{\mathrm{LC}_\sigma(g_i)} q_i \, \mathrm{LT}_\sigma(g_i)$.

Finally, to prove uniqueness, suppose there exist other polynomials $q'_1, \ldots, g'_s$ and r' which satisfy conditions (a), (b), and (c) such that $f = q'_1 g_1 + \cdots + q'_s g_s + r'$. Then we have

$$0 = (q_1 - q'_1) g_1 + \cdots + (q_s - q'_s) g_s + (r - r') \qquad (*)$$

Now condition (a) implies that $\mathrm{LT}_\sigma(r - r') \notin \langle \mathrm{LT}_\sigma(g_1), \ldots, \mathrm{LT}_\sigma(g_s) \rangle$, and condition (c) implies that for each $i \in \{1, \ldots, s\}$,

$$\mathrm{LT}_\sigma((q_i - q'_i) g_i) \notin \langle \mathrm{LT}_\sigma(g_1), \ldots, \mathrm{LT}_\sigma(g_{i-1}) \rangle \text{ with } q_i \neq q'_i.$$

Thus the leading term with respect to σ of the summands in (*) are pairwise different from those of smaller index. This is possible only when $(q_1 - q'_1) = \cdots, = (q_s - q'_s) = (r - r') = 0$. This completes the proof. $\square$

Definition 2.3.20. Let $f, g_1, \dots, g_s \in A_n \setminus \{0\}$, and let $\mathscr{G}$ be the tuple $(g_1, \dots, g_s)$. Let the representation

$$f = q_1 g_1 + \dots + q_s g_s + r$$

be obtained by applying left Division Algorithm on the polynomial f and the tuple $\mathscr{G}$. Then the Weyl polynomial $r \in A_n$ is called the **left normal remainder** of f with respect to $\mathscr{G}$ and is denoted by $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$, or simply by $\mathrm{NR}_{\mathscr{G}}(f)$ if no confusion can arise. Moreover, we have $\mathrm{NR}_{\mathscr{G}}(0) = 0$.

The normal remainder r of a polynomial $f \in A_n$ with respect to an s-tuple $\mathscr{G} = (g_1, \dots, g_s)$ of polynomials depends greatly on the *ordering* of the tuple $\mathscr{G}$. This can be seen in the following example.

Example 2.3.21. Consider the Weyl Algebra $A_1 = \mathbb{Q}[x_1, \partial_1]$ and let the term ordering be $\sigma =$ `DegRevLex`. Let $g_1 = x_1^3 \partial_1^3 - 5x_1 \partial_1 - 1$, $g_2 = x_1^2 \partial_1^4 + 2\partial_1^3$, and $f = x_1^4 \partial_1^5 - 4x_1 \partial_1^3 - 4\partial_1^3$. Now if $\mathscr{G} = (g_1, g_2)$, then the left Division Algorithm 2.3.18 gives

$$\mathrm{NR}_{\sigma,\mathscr{G}}(f) = 17x_1^2\partial_1^3 - 4x_1\partial_1^3 - 19x_1\partial_1^2 - 4\partial_1^3 - 36\partial_1$$

whereas if $\mathscr{G} = (g_2, g_1)$, then $\mathrm{NR}_{\sigma,\mathscr{G}}(f) = 0$.

This ordering of the elements in the tuple can also affect the number of steps required by Algorithm 2.3.18 to complete the computation. But if we follow the Division Algorithm exactly the way as stated, that is, for a fixed ordered tuple, the output of the algorithm is uniquely determined as proved in part (d) of Proposition 2.3.19. Of course, the output also depends on the choice of the term ordering σ on A_n. On the other hand, as in the commutative case, the Division Algorithm has very nice properties when it is applied to Gröbner bases. More precisely, let f be a Weyl polynomial of a left ideal $I \subset A_n$ and let the set $G = \{g_1, \dots, g_s\}$ be a left Gröbner basis of I with respect to a term ordering σ on A_n. Let $\mathscr{G} = (g_1, \dots, g_s)$. Then the normal remainder, $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ is always unique no matter how the tuple $\mathscr{G}$ is ordered (see Theorem 2.4.1).

Remark 2.3.22. In the above setting, the normal remainder $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ of a polynomial $f \in A_n$ is referred to as **normal form** of f with respect to the ideal I and the term ordering σ and is denoted by $\mathrm{NF}_{\sigma,I}(f)$ or simply by $\mathrm{NF}_{\sigma}(f)$ if it is clear which ideal is considered. The normal form $\mathrm{NF}_{\sigma,I}(f)$ of $f \in A_n$ with respect to the ideal $I \subset A_n$ is the unique element of A_n with the property that $f - \mathrm{NF}_{\sigma,I}(f) \in I$. In particular, it does not depend on the particular σ-Gröbner basis chosen. (see [27] Proposition 2.4.7).

Definition 2.3.23. Let σ be a term ordering on A_n and let $f,g \in A_n$ be two Weyl polynomials in standard form. Let $\mathrm{LT}_\sigma(f) = x^\alpha\partial^\beta$ and $\mathrm{LT}_\sigma(g) = x^\gamma\partial^\delta$. Let $t_{fg} = \frac{\mathrm{lcm}(\mathrm{LT}_\sigma(f),\mathrm{LT}_\sigma(g))}{\mathrm{LT}_\sigma(f)} \in B_n$. We define **S-polynomial** of f and g to be the standard form of the Weyl polynomial $S_{fg} \in A_n$ given by

$$S_{fg} = \frac{t_{fg}}{\mathrm{LC}_\sigma(f)} f - \frac{t_{gf}}{\mathrm{LC}_\sigma(g)} g \tag{2.3}$$

Note that $S_{gf} = -S_{fg}$ and S_{fg} belongs to the left ideal generated by f,g. Thus, $S_{fg} \in I$ where I is a left ideal generated by a set F such that $f,\, g \in F$.

With these definitions of the term ordering, S-polynomials, and the normal remainder algorithm, the Gröbner basis of an ideal $I \subset A_n$ can now be obtained in an analogous way to the well-known commutative case. Below we present the **left Buchberger algorithm** for computing Gröbner basis of a left ideal $I \subset A_n$ with respect to a term ordering σ.

Algorithm 2.3.24. *The Left Buchberger Algorithm*: `LWGB`(I)

`Input:` Ideal $I := \langle f_1,\dots,f_s\rangle$ of A_n and a term ordering σ.
`Output:` A Gröbner basis for I with respect to σ
$B := \{(f_i,f_j)\,|1 \le i < j \le s\}$
$\mathscr{G} := (f_1,\dots,f_s)$
while $(B \neq \emptyset)$ **do**
  Take any pair (f,f') from the set B
  $B := B \setminus \{(f,f')\}$
  $h := S_{ff'}$
  $r := \mathrm{NR}_{\sigma,\mathscr{G}}(h)$
  if $(r \neq 0)$ **then**
    $B := B \cup \{(g,r)\,|g \in \mathscr{G}\}$
    $\mathscr{G} := \mathscr{G} \cup \{r\}$
  end if
end while
return $\mathscr{G}$

Theorem 2.3.25. *Let $G = \{g_1, \ldots, g_s\}$ be a finite subset of the Weyl algebra A_n and let σ be a term ordering.*

(1) The set G is a left σ-Gröbner basis of the ideal $I = \langle G \rangle$ if and only if the normal remainder of every S-polynomial $S_{g_i g_j}$ $(i \neq j)$ with respect to $(g_1, \ldots, g_s)$ is 0.

(2) The Left Buchberger Algorithm 2.3.24 terminates and returns a Gröbner basis of the ideal I with respect to σ.

Proof. The proof is similar to the commutative case, for instance see [27] Theorem 2.5.5. □

The study of optimizations of Buchberger's Algorithm for maximum speed is an active research area both in the commutative and the non-commutative settings. Not all the optimizations of Buchberger's Algorithm in the commutative ring $P = K[x_1, \ldots, x_n]$ are true in the setting of the Weyl algebra A_n. For example, the *coprimality test* (see [27], Cor. 2.5.10) does not hold in general for Weyl algebras. This test states that, if $G = \{g_1, \ldots, g_s\} \subset P \setminus \{0\}$ generates the ideal $I = \langle g_1, \ldots, g_s \rangle$ and if the leading terms of the elements $g_1, \ldots, g_s$ are pairwise coprime then G is a σ-Gröbner basis of I. This is not true in general for Weyl algebras. For example, consider the Weyl algebra $A_1 = \mathbb{Q}[x_1, \partial_1]$ and $g_1 = x_1, g_2 = \partial_1$. Let I be an ideal generated by the set $G = \{g_1, g_2\}$. Then this criterion would imply that G is a Gröbner basis of I which is of course not true since $g_2 g_1 - g_1 g_2 = 1$. However, for Weyl algebras, one of the optimizations of the Left Buchberger's Algorithm is possible by using a similar criterion which is known as *Generalized Product Criterion*. It is explained in [30] (Ch. 2, Lemma 4.11).

Definition 2.3.26. Let $I = \langle f_1, \ldots, f_r \rangle$ be an ideal of the Weyl algebra A_n and let σ a term ordering. Let $d \geq \max\{\deg(f_1), \ldots, \deg(f_r)\}$. Let H be the output of the left Buchberger Algorithm, modified so that each computation involving polynomials of degree higher than d is not performed. The set H then contains polynomials of degree less than or equal to d and it is called a **left partial Gröbner basis** of the ideal I with respect to the term ordering σ and the degree d is called the **degree bound** for this partial Gröbner basis H.

Remark 2.3.27. Note here that if G is a left σ-Gröbner basis of a left ideal $I \subset A_n$, then it does not mean that a left partial σ-Gröbner basis H with degree bound d, necessarily contains all Gröbner basis elements $g \in G$ such that $\deg(g) \leq d$. It should be clear from the above definition that H is computed by interrupting the left Buchberger Algorithm to skip any operation involving polynomial of degree higher than d. That is, if the process is allowed to continue from the interruption point, then it might be possible that new Gröbner basis elements have degree less than or equal to the degree bound d of the partial Gröbner basis H.

2.4 Left Ideal Membership

Among many applications of Gröbner bases of ideals, we are mainly interested in the **left ideal membership problem**. That is, given a left ideal $I \subset A_n$ and a Weyl polynomial $f \in A_n$, the ideal membership problem is to decide whether $f \in I$. Even in the commutative setting, the ideal membership problem is EXPSPACE-hard. In particular, this implies that it is in neither NP nor co-NP (see [36] or [53]). Just like in the commutative case (see [27]), the solution to this problem for left ideals in Weyl algebras is provided by the following theorem.

Theorem 2.4.1. *Let I be a non-zero left ideal of a Weyl algebra $A_n = K[x, \partial]$ and let $G = \{g_1, \dots, g_r\}$ be a finite subset of A_n. Let σ be a term ordering on A_n and let $\mathscr{G} = (g_1, \dots, g_r)$. Then the following are equivalent*

(1) G is a left σ-Gröbner basis for I.

(2) For $f \in A_n$, we have $f \in I$ if and only if $\mathrm{NR}_{\sigma,\mathscr{G}}(f) = 0$

(3) Every $f \in I$ has a standard (left) representation with respect to G. That is, there exist $\ell_1, \dots, \ell_r \in A_n$ such that $f = \ell_1 g_1 + \cdots + \ell_r g_r$ and $\mathrm{LT}_\sigma(\ell_j g_j) \leq \mathrm{LT}_\sigma(f)$ for all j such that $\ell_j g_j \neq 0$.

(4) For any Weyl polynomial $f \in A_n$, the normal remainder $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ agrees with $\mathrm{NF}_{\sigma,I}(f)$. In particular, the normal remainder does not depend on the order of elements $g_1, \dots, g_r$.

Proof. For parts (1) – (3), see [30], Theorem 1.16. Part (4) is similar to the commutative case, see [27], Corollary 2.4.9. □

The part (2) of this theorem provides us a way of deciding left ideal membership in two steps. That is, given a left ideal $I \subset A_n$ and a Weyl polynomial $f \in A_n$, we can decide ideal membership of f as follows:

(a) Compute a left σ-Gröbner basis $G = \{g_1, \ldots, g_s\}$ of the ideal I and let $\mathscr{G} = (g_1, \ldots, g_s)$

(b) Compute the normal remainder $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ by using the normal remainder algorithm with respect to $\mathscr{G}$. If $\mathrm{NR}_{\sigma,\mathscr{G}}(f) = 0$, then $f \in I$, otherwise $f \notin I$.

Remark 2.4.2. Here we note that the complexity of deciding left ideal membership depends on the complexity of the computation of Gröbner bases of left ideals in Weyl algebra and secondly on the computation of normal remainders of Weyl polynomials. The degree bound for Gröbner bases in Weyl algebras is established to be doubly-exponential (see [5] for details). Regardless of possible optimizations of Buchberger's Algorithm (2.3.24) for computing Gröbner bases of ideals in Weyl algebras, we observe that Weyl multiplication (see 2.1.5) makes the computation harder by increasing the size of polynomials and hence memory consumption for storing intermediate results during the computation. In fact this slows down the reduction process of computing the normal remainder (see Algorithm 2.3.18) with respect to a tuple $\mathscr{H}$ of Weyl polynomials, especially when $\mathscr{H}$ is not a Gröbner basis.

The following proposition will be useful in choosing a polynomial in an ideal I of A_n.

Proposition 2.4.3. *Consider a Weyl algebra $A_n = K[x_1, \ldots, x_n, \partial_1, \ldots, \partial_n]$ and a term ordering σ. Let I be a left ideal of A_n and let $G = \{g_1, \ldots, g_s\}$ be its left σ-Gröbner basis. For an arbitrary polynomial $f \in A_n$, the polynomial $g = f - \mathrm{NR}_{\sigma,\mathscr{G}}(f)$ belongs to the ideal I, where $\mathscr{G} = (g_1, \ldots, g_s)$.*

Proof. The proof follows immediately from Theorem 2.4.1. $\square$

This concludes our brief overview. Further results about Gröbner bases in Weyl algebras will be recalled as needed.

2.5 Constructing Gröbner Bases of Left Ideals of A_n

Because of the relation $\partial x = x\partial + 1$, it is very likely that an ideal generated by a set of randomly chosen Weyl polynomials contains 1 and hence has a Gröbner basis equal to $\{1\}$. For example, in the Weyl algebra $A_1 = \mathbb{Q}[x,\partial]$, the following ideals are trivial ideals:
$\langle x,\partial\rangle$, $\langle 2x^2+\partial,\partial\rangle$, $\langle x^2+x\partial-\partial, x^3\partial+x\partial-1\rangle$, $\langle x^4\partial^7+x^4, x^9\partial^3+x^2\partial^2-1\rangle$
Likewise, in $A_2 = \mathbb{Q}[x_1,x_2,\partial_1,\partial_2]$ the ideals $\langle x_1^4\partial_1^7-1, x_2^3\partial_2^3+x_1\partial_1+1\rangle$, $\langle x_1^3\partial_1^7+\partial_1-1, x_2^3\partial_2^3+x_1\partial_1+1\rangle$, $\langle x_2^2\partial_1^2-1, x_1\partial_1+\partial_1\rangle$, and $\langle \partial_2^3+x_1\partial_2-1, x_1\partial_1+\partial_1\rangle$ are trivial ideals. Similarly in A_n, $n>1$, it is very likely that after a large amount of computation, the Gröbner basis of an ideal generated by a set of randomly chosen Weyl polynomials turns out to be $\{1\}$. In this section, we propose some ways of finding non-trivial left ideals of the Weyl algebra A_n. For this, let us collect some useful observations.

Proposition 2.5.1. *Let σ be a term ordering on B_n. Let $g \in A_n \setminus \{0\}$ and let $I = \langle g\rangle$ be the left principal ideal generated by g. Then $G = \{g\}$ is a left σ-Gröbner basis of I.*

Proof. This claim is an immediate consequence of the Proposition 2.3.12. □

Claim in this Proposition means that for a Weyl polynomial $g \in A_n \setminus \{0,1\}$ the left principal ideal $I = \langle g\rangle$ is a non-trivial ideals of the Weyl algebra A_n. The following proposition gives us a way of constructing non-trivial ideals of the Weyl algebra A_n that are not principal.

Proposition 2.5.2. *Let $A_n = K[x_1,\dots,x_n,\partial_1,\dots,\partial_n]$ be Weyl algebra of index n over a field K and let σ be a term ordering on A_n. Let $G = \{g_1,\dots,g_r\}$ be such that g_i is a Weyl polynomial in the indeterminates x_i and ∂_i for $i = 1,\dots,r$. Then the ideal $I = \langle g_1,\dots,g_r\rangle$ is a non-trivial left ideal of A_n. In fact, the set G is a left σ-Gröbner basis of the ideal I.*

Proof. Note that for all i,j, we have $g_i \cdot g_j = g_j \cdot g_i$, *i.e.* g_i and g_j commute for all i,j. Moreover, by construction, the leading terms of the elements $g_1,\dots,g_r$ are pairwise coprime. Therefore, the claim follows from the commutative Product Criterion (see [27], Corollary 2.5.10). □

Using this proposition, we can construct non-trivial ideals in Weyl algebras as follows:

Example 2.5.3. Consider the Weyl algebra $A_2 = K[x_1, x_2, \partial_1, \partial_2]$ of index 2 over the base field $K = \mathbb{F}_{31}$ and let the term ordering be $\sigma = \texttt{DegRevLex}$. Let $I = \langle g_1, g_2 \rangle$ be given by

$$\begin{aligned} g_1 &= 17x_1^3\partial_1^4 + 21x_1^2\partial_1^3 - 3x_1^2 - 2\partial_1^2 + 14x_1\partial_1 + 12x_1 - 13\partial_1 - 21 \\ g_2 &= 11x_2^3\partial_2^4 + 21x_2^2\partial_2^2 + 25x_2^2 - 30\partial_2^2 + 21x_2^2 - 7x_2\partial_2 - 3 \end{aligned}$$

Then the ideal I is a left ideal of A_n and the set $G = \{g_1, g_2\}$ is its left σ-Gröbner basis.

Example 2.5.4. Consider the Weyl algebra $A_4 = K[x_1, x_2, x_3, x_4, \partial_1, \partial_2, \partial_3, \partial_4]$ of index 4, over the base field $K = \mathbb{F}_3$, and let the term ordering be $\sigma = \texttt{DegRevLex}$. Let $I = \langle g_1, g_2, g_3 \rangle$ be given by

$$\begin{aligned} g_1 &= x_1^6\partial_1^5 + 2x_1^2\partial_1^4 - x_1^2 - \partial_1 - 1 \\ g_2 &= x_2^5\partial_2^6 + x_2^2\partial_2^4 + \partial_2^2 - x_2 + \partial_2 + 1 \\ g_3 &= x_3^3x_4^3\partial_3^2 - 2x_4\partial_3 + \partial_3\partial_4 + x_3 - x_4 + \partial_4 + 1 \end{aligned}$$

Then the ideal I is a left ideal of A_n and $G = \{g_1, g_2, g_3\}$ is a left σ-Gröbner basis of I.

Remark 2.5.5. Recall that Weyl a polynomial $f \in C_n$ commutes with every element of the Weyl algebra A_n when the base field K has positive characteristic p. Now consider the Weyl algebra $A_1 = K[x, \partial]$ with the base field $K = \mathbb{F}_p$ of positive characteristic p and let σ be a term ordering on B_1. We can now create a non trivial left ideal I of A_1 generated by two Weyl polynomials f_1 and f_2 as follows: Choose a polynomial $f_1 \in C_n \setminus \{1, 0\}$ and set $f_2 \in A_1 \setminus C_n$ such that $\mathrm{NR}_{\sigma, f_2}(f_1) \notin \mathbb{F}_p \setminus \{0\}$. Then there is a very high probability that the ideal $I = \langle f_1, f_2 \rangle$ is a non-trivial left ideal of A_1. That is, I constructed this way will rarely be a trivial ideal. Moreover, if for the generating polynomial f_2, $\mathrm{LT}_\sigma(f_2) = x^\alpha \partial^\beta$ is such that both $\alpha, \beta \geq 2$, then it will be very likely that minimum number of elements in any left σ-Gröbner basis are more than 2. Here it does not mean that if the polynomials f_1, f_2 are not selected as suggested above then the ideal I cannot be a non-trivial ideal of A_n. For instance, the ideal $I = \langle x^7+1,\ x\partial^2+x^2+x+1\rangle$ of $A_1 = \mathbb{F}_2[x,\partial]$ with the term ordering $\sigma =$ `DegRevLex` is a non trivial left ideal of A_1 and its reduced left Gröbner basis G contains 3 polynomials as given below,

$$G = \{\partial^6+x^4+\partial^4+x^3, \quad x^5+\partial^4+x^2+1, \quad x\partial^2+x^2+x+1\},$$

whereas if $I = \langle x^7+1, x^2\partial+x^2+x+1\rangle$ then we have $G = \{1\}$. In fact, we suggested above technique to minimize the probability of getting a trivial Gröbner basis $G = \{1\}$ of a properly chosen ideal I.

We illustrate the technique described in Remark 2.5.5 in the following example.

Example 2.5.6. Consider the Weyl algebra $A_1 = \mathbb{F}_7[x,\partial]$ over the field $\mathbb{F}_7$ of characteristic 7 and let $\sigma =$ `DegRevLex`. Take $f_1 = \partial^7-1$, $f_2 = x^3\partial^3+x^2\partial-\partial-1$ then $I = \langle f_1, f_2\rangle$ is a non-trivial left ideal of A_1 and a left Gröbner basis G of the ideal I consists of 7 polynomials[1] respectively having 19, 21, 19, 18, 17, 17, and 4 terms . Note here that $f_1 \in C_1 = \mathbb{F}_7[x^7,\partial^7]$.

Using the technique described in Remark 2.5.5, we can construct non-trivial ideals of Weyl algebras of any index $n > 1$. We illustrate this by the following example.

Example 2.5.7. Consider the Weyl algebra $A_2 = K[x_1,x_2,\partial_1,\partial_2]$ of index 2 over the field $K = \mathbb{F}_3$ and let $\sigma =$ `DegRevLex`. Let I be the ideal of A generated by the following Weyl polynomials

$$\begin{aligned}
f_{11} &= x_1^3\partial_1^3-1\\
f_{12} &= x_1^2\partial_1+x_1-\partial_1+1\\
f_{21} &= x_2^6\partial_2^6+x_2^3\partial_2^3+\partial_2^3-1\\
f_{22} &= x_2^2\partial_2^2-x_2\partial_2^2+x_2^2+1
\end{aligned}$$

Then the ideal I is a non-trivial ideal of A_2 and its reduced σ-Gröbner basis is the set $G = \{g_1,\ldots,g_8\}$ of 8 Weyl polynomials where

$$\begin{aligned}
g_1 = {} & \partial_2^7-x_2\partial_2^5+x_2^5-x_2^4\partial_2-x_2^4-x_2\partial_2^3+\partial_2^4+x_2^3+x_2^2\partial_2+x_2\partial_2^2-\partial_2^3-\\
& x_2^2+x_2\partial_2+\partial_2^2+1,
\end{aligned}$$

[1]These Gröbner basis elements are given in the Appendix C.1

$$\begin{aligned}
g_2 &= x_2^6 - x_2^5 - x_2\partial_2^4 + \partial_2^5 + x_2^3\partial_2 - x_2\partial_2^3 - x_2^3 - x_2\partial_2^2 - \partial_2^3 + x_2^2 + \partial_2^2 - \\
&\quad x_2 + \partial_2 - 1, \\
g_3 &= x_2\partial_2^6 - x_2^5 - x_2\partial_2^4 - \partial_2^5 - x_2\partial_2^3 - \partial_2^4 - x_2^2\partial_2 + \partial_2^3 + x_2^2 + x_2\partial_2 - x_2 - \\
&\quad \partial_2 + 1, \\
g_4 &= x_1^3 + x_1^2 - x_1\partial_1 - x_1 + \partial_1 + 1, \\
g_5 &= \partial_1^3 + x_1^2 - x_1\partial_1 - x_1 + \partial_1 + 1, \\
g_6 &= x_1\partial_1^2 + x_1^2 - \partial_1^2 - \partial_1 - 1, \\
g_7 &= x_2^2\partial_2^2 - x_2\partial_2^2 + x_2^2 + 1, \\
g_8 &= x_1^2\partial_1 + x_1 - \partial_1 + 1.
\end{aligned}$$

Note that the polynomials f_{11} and f_{21} belong to the center of the Weyl algebra A_2. Similarly, a left σ-Gröbner basis of the ideal generated by $f_{11} = x_1^3\partial_2^3 - 1$, and $f_{12} = x_1^2\partial_1^2 + x_2 + \partial_2 + 1$ consists of the following 5 polynomials:

$$\begin{aligned}
g_1 &= x_2^3\partial_2^6 + \partial_2^9 - x_2^2\partial_2^6 + x_2\partial_2^7 - \partial_2^8 + x_2\partial_2^6 + \partial_2^7 + \partial_1^6 - \partial_2^6, \\
g_2 &= x_1\partial_1^4 - x_2^2\partial_2^3 + x_2\partial_2^4 - \partial_2^5 - x_2\partial_2^3 - \partial_2^4 + \partial_1^3 - \partial_2^3, \\
g_3 &= x_1x_2\partial_2^3 + x_1\partial_2^4 + x_1\partial_2^3 + \partial_1^2, \\
g_4 &= x_1^3\partial_2^3 - 1, \\
g_5 &= x_1^2\partial_1^2 + x_2 + \partial_2 + 1.
\end{aligned}$$

Later, in chapter 4, we shall use these simple ways of creating ideals in Weyl algebras for constructing hard instances of our proposed cryptosystem.

2.6 Computer Algebra Systems

In order to present our work on Gröbner Bases cryptosystems, we have to perform explicit calculations with Weyl polynomials and to compute Gröbner bases of certain classes of ideals in Weyl algebras. For this purpose and to conclude our work, we have to rely on available computer algebra systems that are designed for computations in Weyl algebras. Most of the time we need an efficient implementation of Buchberger Algorithm 2.3.24 to compute complete as well as partial Gröbner bases of some interesting ideals of Weyl algebras and the Division Algorithm 2.3.18 to compute the normal remainders of Weyl polynomials of very large size with respect to these Gröbner bases. These algorithms and many of their applications have been implemented in several readily available **computer algebra systems** (CAS). The most important CAS available for performing efficient computations with Weyl algebras are presented below:

(1) **Singular**
The CAS Singular [22] is designed for polynomial computations both in commutative and non-commutative algebras and can also be used for working with algebraic geometry and singularity theory. Its powerful package `Plural`, written by V. Levandovskyy (see [30, 31]), provides many algorithms for efficient computations with certain non-commutative algebras. Many of its non-commutative functions are available for computations in Weyl algebras. In particular, we are interested in the following functions for carrying out calculations related to this work:

```
Weyl(), groebner(), slimgb(), std(), twostd(),
NF(), options()
```

For the parameters, syntax and examples related to these functions, we refer to the Singular online manual and to [22].

(2) **Macaulay2**
Macaulay2 is a software system developed by Daniel R. Grayson and Michael E. Stillman [21], for computations in commutative algebra and algebraic geometry. Its package `Dmodules` [32], written by A. Leykin and H. Tsai, contains efficient implementations for working with Weyl algebra and D-modules. Among many, some of the functions that we found useful for our work are: `ideal()`, `gb()`, and ‘`%`’ (an operator used for computing normal remainders).

(3) **Risa /Asir**
Risa/Asir is an open source general computer algebra system written by Noro et. al. [40]. Besides commutative rings, it also provides functions for computing Gröbner bases of ideals in Weyl algebras.

(4) **CoCoA / ApCoCoA**
This CAS [4] is developed and maintained by the teams of L. Robbiano in Genova (Italy) and M. Kreuzer in Passau (Germany). It was initially designed to perform special computations in commutative algebra like computation of border bases and Gröbner bases in commutative rings. ApCoCoA is based on the computer algebra system CoCoA [11]. The ApCoCoA library contains several packages for working with non-commutative algebras and group rings. Our own package `Weyl` has been especially designed to carry out the research work presented in this thesis and to perform many computations in Weyl algebras. The functions available in this package for working with the Weyl algebras are explained in Appendix A.

Note. Through out the thesis, we will refer to one or some of the above CAS for describing our computational results obtained on our 'computing machine', that is, the computer system with 24 GB of RAM, and having Processor: AMD Dual Opteron 2.4 GHz. All computations are performed on this computing machine and therefore all the timings are given accordingly.

Chapter 3

Gröbner Basis Cryptosystems

This chapter is about some preliminary material on cryptography with emphasis on a class of public key cryptosystems known as Gröbner Basis Cryptosystems. In particular, we shall discuss an algebraic public key cryptosystem, the Polly Cracker and its generalization, the commutative Gröbner bases cryptosystem. We describe various known standard attacks for the cryptanalysis of these cryptosystems in the commutative setting. We conclude the chapter by describing a more general class of such cryptosystems that are based on Gröbner bases of modules over certain non-commutative rings and hence develop a base and motivation for our new algebraic public key cryptosystem that is based on Gröbner bases in Weyl algebras and introduced in Chapter 4

3.1 Cryptography

In this section we briefly describe *cryptography* and the basic components of a modern cryptosystem with emphasis on *public key cryptography*. There are many good references on the subject and among them we refer to [38], [9], and [25]. *Cryptology* is the science of secret communication. Using the science of cryptology, the two parties, usually known as *Alice* and *Bob*, can share information on a public network. That is, it is all about secret and secure communication through insecure channels. This process of secret communication means converting original messages or data into secret codes for transmission over a public network. The original message is called '*plaintext*' and the corresponding converted message is known as '*ciphertext*'. When *Alice* wants to send a *'plaintext'* to *Bob*, she converts it into the corresponding *'ciphertext'* via an ***encryption algorithm***. After *Bob* has received the *'ciphertext'* through a public network, he decrypts it back to the *'plaintext'* via a ***decryption algorithm***.

This science is classified into the following two main areas:

(1) **Cryptography** is the part that deals with the designing of a system, known as a ***cryptosystem***, for the encryption and decryption of the data.

(2) **Cryptanalysis** is the part that deals with the breaking of such a cryptosystem and hence checking its security from various directions.

Cryptosystems have been in use since ancient times. In fact, Julius Caesar is said to have used the '*shift cipher*' for secret communication with his generals. In modern times, such cryptosystems have 'no security'. One can use computers to break the encryption scheme by trying all 'possible shifts' in a very short time. Therefore, for designing a truly secure cryptosystem, we should have to consider an other important third character in the process of secret sharing, the eavesdropper usually known as *Eve*. That is, a cryptosystem that *Alice* and *Bob* are using for secret communication should be such that *Eve* is unable to break the system by using her complete potential. The process of an attempt for breaking a system will be called an **attack** on the system.

A typical cryptosystem has following four basic components:

(1) The ***message space*** M, is the set of all possible '*plaintext*' messages.

(2) The ***ciphertext space*** C, is the set of all possible encrypted messages, 'ciphertexts'.

(3) The ***encryption algorithm*** E, a function that maps 'plaintext' into its 'ciphertext'.

(4) The ***decryption algorithm*** D,a function that maps 'ciphertext' back to its corresponding 'plaintext'.

Following are the two major cryptographic methods that have been used in modern cryptosystems:

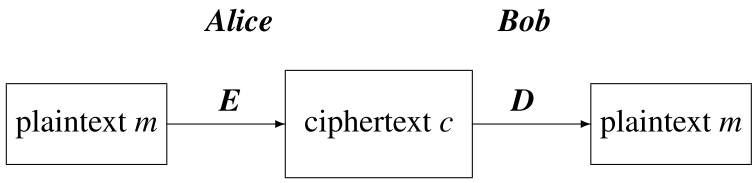

Figure 3.1: cryptosystem

(1) **Secret Key Cryptography** (SKC)
where both parties share a common secret key for encryption and decryption processes (such as **DES** and **AES**).

(2) **Public Key Cryptography** (PKC)
where each party has its own secret key (such as **RSA** (Rivest - Shamir - Adleman) and **El Gamal**)

Cryptosystems in **SKC** and **PKC** are respectively known as *Symmetric Systems* and *Asymmetric Systems*. Although *symmetric systems* are usually more efficient and faster, they have many drawbacks like *security* and *key-management*. The major drawback of these methods is the 'sharing of secret key', that is, SKC requires the prior communication of the secret key between *Alice* and *Bob*. Moreover, if *Alice* has to communicate with n independent parties, she would have to take care of n different 'secret keys' from all the parties. All these keys need to be shared through a trusted and secure channel and should be saved properly. In practice, this may be very difficult to achieve in the modern world of computers. In order to resolve such issues, the introduction of **PKC**, or *asymmetric systems* have played an important role in modern cryptography.

The idea of public key cryptography was first put forward by Whitfield Diffie and Martin Hellman [14] in 1976. They introduced an encryption scheme based on the intelligent idea of not using 'one' single secretly shared key for both encryption and decryption and opened the doors of new world of modern cryptography. In the world of **PKC**, the recipient *Bob* has a key with two parts, namely, a **public key** Q which is published to use by every one and a **secret key** which is kept *secret*. When *Alice* wishes to send data to *Bob*, she uses *Bob*'s public key to encrypt the 'plaintext' via an **encryption rule** e_Q and then *Bob* uses his secret key to decrypt the 'ciphertext' via a **decryption rule** d_Q. The idea behind a public key cryptosystem is that it might be possible to find a cryptosystem where it is computationally infeasible to determine d_Q given e_Q.

At the heart of this concept is the idea of using *one-way function* for encryption. Recall that, a function that is easy to compute but hard to invert is often called a **one-way function**. That is, a one-to-one function $f : X \to Y$ is "one-way" if it is easy to compute $f(x)$ for any $x \in X$ but hard to compute $f^{-1}(y)$ for most randomly selected y in the range of f. Although there are many injective functions that are believe to be "one-way", unfortunately, currently there do not exist such functions that can be proved to be one-way. Of course, the encryption rule e_Q, should not have to be one-way from *Bob*'s point of view because he has to decrypt (invert) the the ciphertext message that he receives in an efficient way. To make the inversion process easier for *Bob*, we use the concept of a *trapdoor* function. A **trapdoor function** is a function that is easy to compute in one direction, yet believed to be difficult to compute in the opposite direction (finding its inverse) without some special information, called the "trapdoor".

Thus it is necessary that *Bob* possesses a *trapdoor*, that is, secret information that permits an easy inversion of e_Q for a given ciphertext. In other word, **PKC** is based on a *trapdoor one-way function*, that is, a one-way function but it becomes easy to invert with the knowledge of certain trapdoor (the "secret" key).

Many public-key cryptosystems have already been proposed and implemented since 1976. Among them, the most important are, **RSA**, *Elliptic-Curve Cryptography* (**ECC**), and the **El Gamal** cryptosystem. The two most commonly used cryptosystems mentioned here, namely **RSA** and **El Gamal**, are respectively based on *integer factorization* and *discrete logarithm* problems. Both problems are considered to be hard to solve for chosen parameters for the corresponding cryptosystem. The drawback of these cryptosystems is that, with the increase in computing power and development of modern computers, the parameters of these cryptosystems need to be modified for achieving a reasonable level of security. The *NP-completeness* or *NP-hardness* of these problems has not been proven yet. In fact, in 1999, *Peter Shor* has discovered a polynomial time algorithm for both the integer-factorization and the discrete logarithm on 'quantum computers'. This motivates researchers to search for cryptosystems that are based on *computationally infeasible* problems. In the next section, we shall describe a *general public-key cryptosystem* **Polly Cracker** that is introduced by Fellows and Koblitz [25] (Chapter 5, §3). The security of this cryptosystem relies on the difficulty of solving a system of algebraic equations. Note that, the problem of 'polynomial system solving' over some finite field is in general an NP-hard problem (see for instance [29]).

3.2 The Polly Cracker Cryptosystems

Before we describe the multivariate algebraic cryptosystem *Polly Cracker*, and in general, the commutative *Gröbner Basis Cryptosystem*, let us first fix some *notation* for subsequent use: Let $P = \mathbb{F}_q[x_1,\ldots,x_n]$ be polynomial ring in n indeterminates over a finite field $\mathbb{F}_q$ with $q = p^e$ for some prime number p and $e > 0$. Let x^α denote $x_1^{\alpha_1}\cdots x_n^{\alpha_n}$, and for $\alpha = (\alpha_1,\ldots,\alpha_n) \in \mathbb{N}^n$, we write $|\alpha| = \alpha_1 + \cdots + \alpha_n$. The elements of the form x^α in P are called **terms**. Let $\mathbb{T}^n$ be the monoid of all terms in P, i.e. $\mathbb{T}^n = \{x^\alpha \mid \alpha = (\alpha_1,\ldots,\alpha_n) \in \mathbb{N}^n\}$. For the basic form of Polly Cracker cryptosystem (PCC), as introduced by Fellows and Koblitz, we assume that the plaintext units are represented as elements of the field $\mathbb{F}_q$. In order to receive a message $m \in \mathbb{F}_q$ from *Alice*, *Bob* chooses his *secret key* by selecting a random element $(a_1,\ldots,a_n) \in \mathbb{F}_q^n$ and his *public key* is the ideal J generated by a set $Q = \{p_1,\ldots,p_s\}$ of polynomials in P such that $p_j(a_1,\ldots,a_n) = 0$, for all $j = 1,\ldots,s$. For sending a message m, *Alice* chooses a random element $\sum_j p_j q_j$ of the ideal J and sends an element $c = m + \sum_j p_j q_j$ to *Bob*. Finally, *Bob* recovers m by evaluating c at $(a_1,\ldots,a_n)$. To sum up we have the following:

Cryptosystem 3.2.1 (Polly Cracker). Let $K = \mathbb{F}_q$ be a finite field, where $q = p^e$ with a prime number p and $e > 0$. Let $P = K[x_1,\ldots,x_n]$ be a commutative polynomial ring. Choose a point $(a_1,\ldots,a_n) \in \mathbb{F}_q^n$. Let I be the ideal generated by $\{x_1 - a_1,\ldots,x_n - a_n\}$. Choose polynomials $p_1,\ldots,p_s \in I$, i.e. for all $i = 1,\ldots,s$,

$p_i(a_1,\ldots,a_n) = 0$. The basic Polly Cracker cryptosystem is then constructed as follows:

(1) **Public key:** A set $Q = \{p_1,\ldots,p_s\}$ of polynomials in P.

(2) **Secret key:** A common zero $(a_1,\ldots,a_n) \in \mathbb{F}_q^n$ of polynomials in Q.

(3) **Message Space:** The message space is $\mathscr{M} = \mathbb{F}_q$, i.e. plaintext units are elements of $\mathbf{F}_q$.

(4) **Ciphertext Space:** The ciphertext units are polynomials in P.

(4) **Encryption:** For encrypting a plaintext message m in $\mathbb{F}_q$, the ciphertext c is computed as:

$$c = m + h_1 p_1 + \cdots + h_s p_s,$$

with suitably chosen $h_1,\ldots,h_s$ in P.

(5) **Decryption:** The Evaluation of c at the common zero $(a_1,\ldots,a_n)$ yields m, i.e. $c(a_1,\ldots,c_n) = m$.

Remark 3.2.2. It is easy for *Bob* to construct a pair $(\mathbf{a}, Q)$, where $\mathbf{a} = (a_1,\ldots,a_n)$ is the secret key in $\mathbb{F}^n$ and Q is the public key. For example, he can randomly choose an $\mathbf{a} \in \mathbb{F}^n$ and arbitrary polynomials h_j, and sets $q_j = h_j - h_j(\mathbf{a})$. On the other hand, for the security of the secret key $\mathbf{a}$, it should be hard to find out common *zero* of public polynomials in Q. Constructing a pair $(\mathbf{a}, Q)$ for a secure system is a non trivial matter. If attacker knows the Gröbner basis $G = \{g_1,\ldots,g_r\}$ of the ideal J generated by the polynomials in the set Q, he can break the cryptosystem by computing normal form of ciphertext c with respect to $\mathscr{G} = (g_1,\ldots,g_s)$.

In [25], *Koblitz* suggested some concrete instances of Cryptosystem 3.2.1 for some combinatorial problems like *Graph 3-Coloring* and *Graph Perfect Code*. The Polly cracker based on such NP-hard problems are also known as *combinatorial-algebraic cryptosystems*. In the next section we shall explain the cryptanalysis of PCC.

3.3 Cryptanalysis of Polly Cracker

Although the Cryptosystem 3.2.1 is based on the NP-hard problem of polynomial system solving over a finite field, it turned out that constructing practically hard instances is very difficult and is an involved task. Note here that, for encrypting a message m, *Alice* has to randomly choose polynomials $h_1, \ldots, h_s \in P$ such that the resulting ciphertext c, should be '*random-looking*'. This choice of polynomials should be such that:

- the ciphertext c should be random-looking
- the message m should be well hidden in the sum $m + p_1 h_1 + \cdots + p_s h_s$.
- monomials/terms used in $h_1, \ldots, h_s$ should not '*shine-through*' the ciphertext c.

For building a concrete instance of the Polly cracker cryptosystem 3.2.1, all these tasks are rather involved. Therefore, a weakly constructed ciphertext can be broken easily with the standard attacks proposed by the cryptanalysts. For details, we refer to ([25], Chapter 5), [48], [49], [23], and [50]. Here, we describe these attacks briefly and later we shall refer to these attacks again to discuss the security of our proposed cryptosystem against these attacks.

3.4 The Chosen Ciphertext Attack

In [48], Steinwandt and Geiselmann describe this attack for the basic Polly Cracker scheme to reveal the secret key and hence completely compromising the security of the encryption scheme. The main assumption of the attack is that the attacker, *Eve*, has temporary access to *Bob*'s decryption black box i.e. *Eve* is able to decrypt the finite number of ciphertext messages that she sends, without actually knowing *Bob*'s secret key. This attack is most serious in the sense that it recovers the complete secret key and hence the attacker can successfully decrypt any stolen ciphertext message. The idea is to send a fake ciphertext to the decryption black box and recover the *Bob*'s original secret key. The attack works as follows:

Attack 3.4.1. *The Chosen Ciphertext Attack*

Assume that, instead of a ciphertext polynomial $c = m + \sum_{j=1}^{s} h_j p_j$, *Alice* sends to *Bob*, a "fake ciphertext",

$$c'_i = x_i + \sum_{j=1}^{s} h_{ij} p_j \text{ with } i = 1, \ldots, s \text{ , and } h_{ij} \in \mathbb{F}_q[X]$$

Then, the specification of Polly Cracker gives no hint on how *Bob* can distinguish such a "fake" ciphertext from a correct one, i.e., from a ciphertext of the form

$$c = m + \sum_{j=1}^{s} h_j p_j \text{ with } m \in \mathbb{F}_q \text{ and } h_1, \ldots, h_s \in P = \mathbb{F}_q[X]$$

Now, the decryption of this fake ciphertext is the evaluation of c'_i at the common zero $(a_1, \ldots, a_n)$, i.e.

$$c'_i(a_1, \ldots, a_n) = x_i(a_1, \ldots, a_n) + \sum_{j=1}^{s} h_{ij} p_j(a_1, \ldots, a_n) = a_i.$$

Hence, learning the plaintext corresponding to c'_i determines the i-th coordinate of the *Bob*'s secret key $\mathbf{a} \in \mathbb{F}_q^n$. Hence learning the plaintexts corresponding to n chosen "fake" ciphertexts $c'_1, \ldots, c'_n$ is enough for *Alice* to reveal the *Bob*'s complete secret key $\mathbf{a}$.

To defeat this attack, it has been suggested to design a decryption algorithm that can recognise "fake" ciphertext messages. For the basic Polly Cracker encryption scheme, there seems to be no straightforward way to recognise fake ciphertext polynomials c'_i as they are valid ciphertexts. Therefore, this encryption scheme is not secure against such kind of attacks.

3.5 The Linear Algebra Attack

In [25], Koblitz explained a linear algebra attack for breaking Cryptosystem 3.7.2 and all its special cases. Basically, the attacker looks for the weaknesses in the construction of the ciphertext c and success of the attack will recover the corresponding plaintext m that *Alice* has sent to *Bob*. The idea of the attack is to reconstruct the polynomials $h_1, \ldots, h_s$ that *Bob* has used for the encryption. The attack works as follows:

In the equation

$$c = m + h_1 p_1 + \cdots + h_s p_s,$$

the eavesdropper, *Eve*, regards the polynomial coefficients $h_1, \ldots, h_s$, respectively, as the polynomials $h'_1, \ldots, h'_s$ of $P = K[X]$ of degree less than or equal to $d = \deg(c) - d_p$, where $d_p = \max\{\deg(p_i), i = 1, \ldots, s\}$ and regards the message constant m as an unknown constant $m' \in \mathbb{F}_q$. She then formulates a linear system of equations using

$$c' = m' + h'_1 p_1 + \cdots + h'_s p_s$$

and then equating the coefficients in c and c'. Let d_o be the initial guess for the degree d_h of the polynomials $h'_1, \ldots, h'_s$ that *Bob* has used for the encryption. To break the Polly Cracker, an attacker has to implement the following attack.

Attack 3.5.1. *The Linear Algebra Attack*

For an instance of the basic Polly Cracker cryptosystem, the **linear algebra attack** works as follows:

`Input:` $c \in P$, $Q = \{p_1, \ldots, p_s\} \subset P$.
`Output:` $m \in \mathscr{M} = \mathbb{F}_q$, the element of the message space $\mathscr{M}$.

(1) Initialize, $d := (\deg(c) - d_p)$.
(2) For $i = 1, \ldots, s$, write the polynomials $h'_i = \Sigma_{|\alpha| \le d} b_{ij} x^\alpha \in P$ with indeterminate coefficients b_{ij}, where $x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ and $|\alpha| = \alpha_1 + \cdots + \alpha_n$. Let m' be the unknown message m, and compute $c' = \sum_i h'_i p_i + m'$.
(3) By equating monomial terms in c and c' formulate a system of linear equations in unknowns b_{ij}, and m'.
(4) Solve the above system of linear equations for finding the values of b_{ij} and m'.

Case-1 If the system has a solution then **return** m'.
Case-2 If system has no solution then

(i) Replace d by $(d+1)$,
(ii) **go to** Step (2).

We illustrate the attack by the following example.

Example 3.5.2. Let us now consider an instance of Polly Cracker with $P = \mathbb{F}_{19}[x_1, x_2]$. Let the public key be $Q = \{p_1, p_2\}$ with

$$\begin{aligned} p_1 &= 7x_1^3x_2 + 6x_1^2 + 4x_1x_2 + x_2^2 + 8x_1 + 2x_2 - 3 \\ p_2 &= -5x_1^3x_2 + 7x_1^2x_2 + 4x_1x_2^2 - 5x_1x_2 + 6x_2^2 + 9x_1 + 4x_2 + 5 \end{aligned}$$

For encrypting the message $m = 8$, let us choose

$$h_1 = -2x_1x_2 + 2x_1 + 5, \qquad h_2 = -x_1 + x_2 + 7.$$

We compute the ciphertext $c = h_1p_1 + h_2p_2 + m$ and get the polynomial

$$\begin{aligned} c = \; & 5x_1^4x_2^2 - 5x_1^3x_2^2 - 5x_1^2x_2^2 + 2x_1x_2^3 - 7x_1^3 + 8x_1^2x_2 - 4x_1x_2^2 + 6x_2^3 - x_1^2 - 6x_2^2 - \\ & 3x_1 + 5x_2 + 9 \end{aligned}$$

Now, for reconstructing the polynomials h_1, h_2 and recovering the message $m = 8$, the attacker, *Eve*, can apply the Attack 3.5 as follows:
By setting $d = 2$ as the initial degree for the polynomials h'_1 and h'_2 and by setting these polynomials as

$$\begin{aligned} h'_1 &= b_{11}x_1^2 + b_{12}x_1x_2 + b_{13}x_2^2 + b_{14}x_1 + b_{15}x_2 + b_{16} \\ h'_2 &= b_{21}x_1^2 + b_{22}x_1x_2 + b_{23}x_2^2 + b_{24}x_1 + b_{25}x_2 + b_{26} \end{aligned}$$

she obtains the general ciphertext polynomial $c' = h'_1p_1 + h'_2p_2 + m_0$ as
$c' = 7b_{11}x_1^5x_2 - 5b_{21}x_1^5x_2 + 7b_{12}x_1^4x_2^2 - 5b_{22}x_1^4x_2^2 + 7b_{13}x_1^3x_2^3 - 5b_{23}x_1^3x_2^3 + 7b_{14}x_1^4x_2 - 5b_{24}x_1^4x_2 + 7b_{21}x_1^4x_2 + 7b_{15}x_1^3x_2^2 - 5b_{25}x_1^3x_2^2 + 7b_{22}x_1^3x_2^2 + 4b_{21}x_1^3x_2^2 + 7b_{23}x_1^2x_2^3 + 4b_{22}x_1^2x_2^3 + 4b_{23}x_1x_2^4 + 6b_{11}x_1^4 + 7b_{16}x_1^3x_2 + 6b_{12}x_1^3x_2 + 4b_{11}x_1^3x_2 - 5b_{26}x_1^3x_2 + 7b_{24}x_1^3x_2 - 5b_{21}x_1^3x_2 + 6b_{13}x_1^2x_2^2 + 4b_{12}x_1^2x_2^2 + b_{11}x_1^2x_2^2 + 7b_{25}x_1^2x_2^2 + 4b_{24}x_1^2x_2^2 - 5b_{22}x_1^2x_2^2 + 6b_{21}x_1^2x_2^2 + 4b_{13}x_1x_2^3 + b_{12}x_1x_2^3 + 4b_{25}x_1x_2^3 - 5b_{23}x_1x_2^3 + 6b_{22}x_1x_2^3 + b_{13}x_2^4 + 6b_{23}x_2^4 + 6b_{14}x_1^3 + 8b_{11}x_1^3 + 9b_{21}x_1^3 + 6b_{15}x_1^2x_2 + 4b_{14}x_1^2x_2 + 8b_{12}x_1^2x_2 + 2b_{11}x_1^2x_2 + 7b_{26}x_1^2x_2 - 5b_{24}x_1^2x_2 + 9b_{22}x_1^2x_2 + 4b_{21}x_1^2x_2 + 4b_{15}x_1x_2^2 + b_{14}x_1x_2^2 + 8b_{13}x_1x_2^2 + 2b_{12}x_1x_2^2 + 4b_{26}x_1x_2^2 - 5b_{25}x_1x_2^2 + 6b_{24}x_1x_2^2 + 9b_{23}x_1x_2^2 + 4b_{22}x_1x_2^2 + b_{15}x_2^3 + 2b_{13}x_2^3 + 6b_{25}x_2^3 + 4b_{23}x_2^3 + 6b_{16}x_1^2 + 8b_{14}x_1^2 - 3b_{11}x_1^2 + 9b_{24}x_1^2 + 5b_{21}x_1^2 + 4b_{16}x_1x_2 + 8b_{15}x_1x_2 + 2b_{14}x_1x_2 - 3b_{12}x_1x_2 - 5b_{26}x_1x_2 + 9b_{25}x_1x_2 + 4b_{24}x_1x_2 + 5b_{22}x_1x_2 + b_{16}x_2^2 + 2b_{15}x_2^2 - 3b_{13}x_2^2 + 6b_{26}x_2^2 + 4b_{25}x_2^2 + 5b_{23}x_2^2 + 8b_{16}x_1 - 3b_{14}x_1 + 9b_{26}x_1 + 5b_{24}x_1 + 2b_{16}x_2 - 3b_{15}x_2 + 4b_{26}x_2 + 5b_{25}x_2 - 3b_{16} + 5b_{26} + m_0$

Equating the corresponding coefficients with the original ciphertext c, she then gets the following system of linear equations in the unknowns $b_{11},\dots,b_{16}, b_{21},\dots,b_{26},m_0$

$$7b_{11}-5b_{21}=0, \qquad 7b_{12}-5b_{22}=5, \qquad 7b_{13}-5b_{23}=0,$$
$$7b_{14}+7b_{21}-5b_{24}=5,$$
$$7b_{15}+4b_{21}+7b_{22}-5b_{25}=-5,$$
$$4b_{22}+7b_{23}=0,$$
$$4b_{23}=0, \qquad 6b_{11}=0,$$
$$4b_{11}+6b_{12}+7b_{16}-5b_{21}+7b_{24}-5b_{26}=0,$$
$$b_{11}+4b_{12}+6b_{13}+6b_{21}-5b_{22}+4b_{24}+7b_{25}=-5,$$
$$b_{12}+4b_{13}+6b_{22}-5b_{23}+4b_{25}=2,$$
$$b_{13}+6b_{23}=0,$$
$$8b_{11}+6b_{14}+9b_{21}=0,$$
$$-3b_{14}+8b_{16}+5b_{24}+9b_{26}=3,$$
$$2b_{11}+8b_{12}+4b_{14}+6b_{15}+4b_{21}+9b_{22}-5b_{24}+7b_{26}=0,$$
$$2b_{12}+8b_{13}+b_{14}+4b_{15}+4b_{22}+9b_{23}+6b_{24}-5b_{25}+4b_{26}=-6,$$
$$2b_{13}+b_{15}+4b_{23}+6b_{25}=6,$$
$$-3b_{11}+8b_{14}+6b_{16}+5b_{21}+9b_{24}=2,$$
$$-3b_{12}+2b_{14}+8b_{15}+4b_{16}+5b_{22}+4b_{24}+9b_{25}-5b_{26}=-4,$$
$$-3b_{13}+2b_{15}+b_{16}+5b_{23}+4b_{25}+6b_{26}=-6,$$
$$-3b_{15}+2b_{16}+5b_{25}+4b_{26}=5,$$
$$-3b_{16}+5b_{26}+m'=9.$$

By solving this system, she then gets

$$b_{11}=0, b_{12}=-2, b_{13}=0, b_{14}=2, b_{15}=0, b_{16}=5,$$

$$b_{21}=0, b_{22}=0, b_{23}=0, b_{24}=-1, b_{25}=1, b_{26}=7,$$

and $m'=8$. This recovers the original message $m=m'=8$ and also the polynomials used for the encryption.

The linear system of equations obtained this way can be easily made infeasible to solve by choosing various parameters as suggested in Notation 3.7.4. For example, as stated in [25] (see Ch. 5 §6), if c and p_i are "sparse" polynomials then method in this general form is exponential time. However, *Koblitz* [25] cited a private communication with H. W. Lenstra Jr. and proposed a modified form of Attack 3.5.1 and call it "intelligent" linear algebra attack.

3.6 Intelligent Linear Algebra Attack

The "intelligent" linear algebra attack was roughly suggested by H.W. Lenstra Jr ([25], Chapter 5). The attack is based on a simple technique of reducing the number of unknowns in the linear system of equations obtained by the linear algebra attack. To explain the attack, we define a set

$$D = \{t \in \mathbb{T}^n \mid \exists t_p \in \bigcup_{i=1}^{s} \mathrm{Supp}(p_i), \text{ s.t. } t \cdot t_p = t_c \text{ for some } t_c \in \mathrm{Supp}(c)\}.$$

Roughly speaking, D is the set of all terms that *Bob* can potentially use for the polynomials $h_1, \ldots, h_s$ in the encryption process. Using this refined form, the attacker proceeds as follows:

Attack 3.6.1. *The "Intelligent" Linear Algebra Attack*

Input: $c \in P$, $Q = \{p_1, \ldots, p_s\} \subset P$.
Output: $m \in \mathscr{M} = \mathbb{F}_q$, the element of the message space $\mathscr{M}$.

(1) Initialize, $d := (\deg(c) - d_p)$.

(2) Compute the set of candidate terms of degree at most d in $h_1 \ldots, h_s$

$$D = \{t \in \mathbb{T}^n \mid \exists t_p \in \bigcup_{i=1}^{s} \mathrm{Supp}(p_i) \text{ s.t. } t \cdot t_p = t_c \text{ for some } t_c \in \mathrm{Supp}(c) \text{ and } \deg(t) \le d\}.$$

(3) Let $h_i' = \sum_{t \in D} b_{ij} t \in P$ with unknown coefficients b_{ij} and let $m' \in \mathbb{F}_q$ be the unknown message m, and compute $c' = \sum_i h_i' p_i + m'$.

(4) By equating monomial terms in c and c' formulate a system of linear equations in unknown coefficients b_{ij}, and the unknown m'.

(5) Solve the above system of equations by using linear algebra.

Case-1 If the system has a solution then return the plaintext message $m = m'$.
Case-2 If the system has no solution then replace d by $d+1$ and go to Step 2.

Remark 3.6.2. Note that the Linear Algebra Attack 3.5.1 will not be feasible if the ciphertext and public polynomials are of very large degree, whereas the intelligent linear algebra attack is very efficient when we have high degree and sparse input polynomials. We also remark here that, the denser the ciphertext polynomial c is, the more difficult the intelligent linear algebra attack will be to apply since this would increase the size of the set D in the Intelligent Linear Algebra Attack 3.6.1. Therefore, we expect to have more unknowns in the linear system obtained in this case.

We illustrate Attack 3.6.1 in the following example using our implementation of the attack in ApCoCoA (see B.2).

Example 3.6.3. Let us apply the intelligent linear algebra attack to the instance of Polly Cracker given in Example 3.5.2, Since $\deg(c) = 6$, we have $d = 6-4 = 2$, and the set D turns out to be

$$D = \{x_1^2,\, x_1x_2,\, x_2^2,\, x_1,\, x_2,\, 1\},$$

containing 6 candidate terms for the polynomials h_1' and h_2'. Therefore, by setting

$$\begin{aligned} h_1' &= b_{11}x_1^2 + b_{12}x_1x_2 + b_{13}x_2^2 + b_{14}x_1 + b_{15}x_2 + b_{16}, \\ h_2' &= b_{21}x_1^2 + b_{22}x_1x_2 + b_{23}x_2^2 + b_{24}x_1 + b_{25}x_2 + b_{26}, \end{aligned}$$

and representing the unknown message by m', we compute $c' = h_1'p_1 + h_2'p_2 + m'$. As explained in Example 3.5.2, by equating monomials terms in c and c', we obtain a linear system of 22 equations in 13 unknowns. By solving this system, we recover the message $m = m' = 8$ by using the package `LinBox` of the CAS ApCoCoA in 0.15 seconds of CPU time on our computing machine.

Note that, the Linear Algebra Attack applied to this instance of Polly Cracker was resulted in a linear system of size 28×13, (see Example 3.5.2) which is almost the same size as we have obtained now by applying the intelligent attack. This is what we have explained in Remark 3.6.2, that when the input polynomials for the attack are dense then the 'intelligent' technique of reducing the number of unknowns for the resulting system of linear equations is not much effective. As we have seen in this example, the set D contains all the terms of degree less than or equal to $d = \deg(c) - d_p = 2$ and therefore, the above linear system has 13 unknowns, 6 for each of h'_1 and h'_2 and one is m' for the message.

To see the effectiveness of this attack, let us now check how the attack works when the input polynomials are sparse. For instance, if we use $h_1 = 2x_1^{21} + 5$ and $h_2 = -x_1^{21} + 7$ for encryption, then the ciphertext polynomial $c = h_1 p_1 + h_2 p_2 + 8$ has degree 24 and $\#\,\mathrm{Supp}(c) = 14$. Therefore, the expected degree $d = 24 - 4 = 20$ for the polynomials h'_1, h'_2 and hence, now the following set

$$D = \{x_1^{20}, x_1^{19}x_2, x_1^{19}, x_1^{18}x_2, x_1^2x_2, x_1x_2^2, x_1^2, x_1x_2, x_2^2, x_1, x_2, 1\}$$

contains 12 candidate terms for each polynomial h_i. Hence, after executing Step (3) and (4) of 'intelligent' Attack 3.6.1, we get a linear system of 43 equations in 25 unknowns. This system has no solution, as the degree $d = 20$ is not sufficient for the polynomials h'_1 and h'_2, since both of the polynomials actually used for the encryption are of degree 21 and there is highest degree term has cancelled in c. As suggested in Step (5-ii), we replace d by $d + 1 = 21$ and this results in addition of three more terms in the set D. That is, this time

$$D = \{x_1^{21}, x_1^{20}x_2, x_1^{19}x_2^2, x_1^{20}, x_1^{19}x_2, x_1^{19}, x_1^{18}x_2, x_1^2x_2, x_1x_2^2, x_1^2, x_1x_2, x_2^2, x_1, x_2, 1\},$$

contains 15 candidate terms for each h'_1 and h'_2. Hence, again after Step (3) and (4), we now get a system of linear equations of size 50×31 and recover the message $m = m' = 8$ in 0.74 seconds of CPU time on our computing machine using CAS ApCoCoA.

In contrast, if we apply the Linear Algebra Attack, as explained in Example 3.5.2, to this instance of Polly Cracker, with $d = 20$, we get a linear system of equations of size 325×463. After replacing d by $d + 1 = 21$, the resulting system of linear equations has size 351×507 and this time we recover the message $m = m' = 8$ in 62.3 seconds of CPU time on our computing machine using CAS ApCoCoA.

Remark 3.6.4. In [25], to defeat Attack 3.6.1, the cited private communication also suggested that *Bob* must carefully build at least one term t' into at least one h_j such that t' times any term in p_j is cancelled in the entire sum $\sum h_j p_j$. Moreover, terms t' with this property should not be too few or easy to guess, since otherwise the cryptanalyst would simply adjoin those terms to D.

The cryptanalysis of some special instances of Polly Cracker cryptosystems is also possible by several other methods of attacks. These attacks are either variants of linear algebra attacks or rely on the structural weaknesses of the Polly Cracker encryption schemes, like evaluation of the polynomials at a common zero. That is, evaluation of polynomials can also leak significant information about the secret key. We refer to [49], [23], and [35] for details on these attacks. In the next section we will describe the generalised form of the Polly Cracker cryptosystem and study its security against these standard attacks.

3.7 Commutative Gröbnr Basis Cryptosystems

The PCC has soon been generalised to Commutative Gröbner Basis Cryptosystems by replacing the underlying NP-hard problem of *polynomial system solving* by the EXPSPACE-hard problem of *computing Gröbner bases of ideals in a commutative polynomial ring*. For the theory of Gröbner basis of ideals in commutative polynomial rings we refer to [27].

Let K be a finite field and let $P = K[x_1, \ldots, x_n]$ be a polynomial ring in n indeterminates over the field K. Using the notation of Section 3.2, let $\mathbb{T}^n$ be the set of all terms in P which form the K-vector space basis of the ring P. The term ordering on $\mathbb{T}^n$ is defined as follows in the setting of P.

Definition 3.7.1. A complete ordering σ on $\mathbb{T}^n$ is called a **term ordering** if it has the following properties:

(1) An inequality $x^{\alpha} <_{\sigma} x^{\alpha'}$ implies

$$x^{\alpha+\alpha''} <_{\sigma} x^{\alpha'+\alpha''}$$

for all $\alpha, \alpha', \alpha'' \in \mathbb{N}^n$.

(2) The ordering σ is *well-founded*, i.e. we have $1 <_\sigma t$ for all $t \in \mathbb{T}^n \setminus \{1\}$.

Then, in the Gröbner basis setting, the secret key is replaced by the Gröbner basis $G = \{g_1, \ldots, g_t\}$ of an ideal $I \subset P$ and the public key is the ideal $J = \langle Q \rangle$ generated by the set $Q = \{p_1, \ldots, p_s\} \subset I$. Further, we denote the complement of the set of leading terms of the ideal I by $\mathscr{O}_\sigma(I)$. The message space $\mathscr{M}$ is then either entire set $\mathscr{O}_\sigma(I)$ or a subset of it. That is, messages are polynomials in P that cannot be reduced modulo the Gröbner basis G. With these ingredients, we define the commutative Gröbner Basis Cryptosystem (**CGBC**) as follows:

Cryptosystem 3.7.2. ***Commutative Gröbner Basis Cryptosystem*:** Let P be a commutative polynomials ring over a field K and let σ be a term ordering on $\mathbb{T}^n$. Let $I \subset P$ be an ideal of P having a Gröbner basis $G = \{g_1, \ldots, g_r\}$ with respect to σ and let $\mathscr{G} = (g_1, \ldots, g_r)$. Then a **CGBC** is constructed as follows:

1. **Public key:** The set $Q = \{p_1, \ldots, p_s\}$ of polynomials in the ideal $I \subset P$ such that the Gröbner basis of the ideal $J = \langle Q \rangle$ is infeasible to compute.

2. **Secret key:** Gröbner basis $G = \{g_1, \ldots, g_r\}$ of the ideal $I \subset P$.

3. **Message Space:** The set $\mathscr{M}$ of all polynomials that cannot be reduced modulo the Gröbner basis G.

4. **Encryption:** The 'plaintext' message $m \in \mathscr{M} \subseteq \mathscr{O}_\sigma(I)$ is encrypted as:

$$c = m + h_1 p_1 + \cdots + h_s p_s$$

   with suitably chosen $h_1, \ldots, h_s$ in P.

5. **Decryption:** The normal remainder of the polynomial c with respect to the tuple $\mathscr{G} = (g_1, \ldots, g_r)$ yields m. That is, $\mathrm{NR}_{\sigma,\mathscr{G}}(c) = m$

Remark 3.7.3. In this setting, again it is very easy for *Bob* to choose a pair (G, Q) for constructing an instance of a CGBC. For example, after choosing a Gröbner basis G of an ideal $I \subset P$, in order to choose the set $Q = \{p_j \mid j = 1, \ldots, s\}$ of polynomials in the ideal I, he can choose for each $j = 1, \ldots, s$, an arbitrary polynomial

$h_j \in P$ and set $p_j = h_j - \mathrm{NR}_G(h_j)$. His public key is then the set $Q \subset I$ of polynomials $p_1, \ldots, p_s$. On the other hand, for the security of CGBC he has to make sure that a Gröbner basis of the ideal J generated by the polynomials in the public key Q should be hard to compute. Of course, it is not the only thing on which the security of CGBC relies. Later we shall see that, as in the case of Polly Cracker cryptosystems, the cryptanalysis of CGBC is also possible by using the attacks where the attacker does not have to compute a Gröbner basis or a complete Gröbner basis.

Notation 3.7.4 (CGBC Parameters:). The Polly Cracker cryptosystem 3.2.1 is a special case of CGBC. In order to use a CGBC, one has to consider following parameters for its construction:

- p, the characteristic of the field K,
- n, the number of indeterminates of the ring P,
- s, the number of polynomials in the public key Q,
- $d_p = \max\{\deg(p_i) \mid p_i \in Q\}$, and
- $d_h = \max\{\deg(h_i) \mid 1 \leq i \leq s\}$, and
- d_c, the degree of the ciphertext c.

Although the security of the Gröbner basis cryptosystems relies on the fact that the computation of Gröbner bases of ideals in commutative polynomial rings is, in general, EXPSPACE-hard (see [53] §21.7). Unfortunately, the cryptanalysis of these cryptosystems can be carried out not only by using the attacks where an attacker does not need to compute a Gröbner basis, but also by using another attack, proposed by T. Mora et. al. [8], where the attacker can compute a successful *partial Gröbner basis*. We describe this attack in the next section. The existence of these attacks prompted T. Mora and others to conjecture that the ideal membership cannot be used to construct a public key cryptosystem. Let us now study the security issues of CGBC against known standard attacks.

Remark 3.7.5. (**Linear Algebra Attacks on CGBC**) For an instance of a CGBC, the *Basic Linear Algebra Attack* 3.5 and the *Intelligent Linear Algebra Attack* 3.6

work exactly the same way as for the basic Polly Cracker cryptosystem with one exception. In this case, instead of representing the plaintext message by an unknown constant $m' \in \mathbb{F}_q$, we let $m' = \sum m_i x^\alpha$ as a polynomial m' of the message space $\mathscr{M}$ with indeterminate coefficients m_i. We can then create the linear system of equations in the unknowns b_{ij} and m_i and recover the plaintext message $m = m'$ by solving that linear system of equations. Again, the basic linear algebra attack can be easily made infeasible to work by appropriately setting the CGBC-parameters n and d_c. Therefore, the only serious linear algebra attack is the "intelligent" Attack 3.6.1. To defeat the attack various suggestions have been proposed (see for instance Remark 3.6.4 and [51]) but there do not exist concrete instances of CGBC where infeasibility of this attack can be checked.

3.8 Attack By Partial Gröbner Basis

In [8], an other attack was proposed for the standard Polly Cracker cryptosystem 3.2.1 and its generalised form CGBC 3.7.2. The idea of the attack is based on a result from [13], cited in [8]. It states that if a polynomial is constructed by adding multiples $h_j p_j$ of elements in an ideal, where the degree of $h_j p_j$ is known to be bounded by D, then in testing Ideal Membership by means of a Gröbner basis one can ignore steps in the algorithm involving polynomials of degree greater than D. This, essentially means the following: Let $I = \langle p_1, \ldots, p_s \rangle$ be the ideal in the commutative polynomial ring P and let σ be a degree compatible term ordering on $\mathbb{T}^n$. If the polynomial $f \in P$ be such that $\deg(f) \leq D$ and $f = \sum_j h_j p_j + \mathrm{NR}_{\sigma,\mathscr{G}}(f)$ with $\deg(h_j p_j) \leq D$ holds. Then for deciding the ideal membership of f in the ideal I, do not compute a Gröbner basis of I, but run Buchberger Algorithm modified to compute $\mathscr{H}$ such that each computation involving polynomials of degree higher than D is not performed. Then the $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ can be computed by reduction of f via the partial Gröbner basis $\mathscr{H}$.

This idea of using a partial Gröbner basis for computing normal remainder can be used for trying to break an instance of CGBC and to reveal the plaintext message m. First, note that the attacker, *Eve* knows the public polynomials $p_1, \ldots, p_s$, the ciphertext polynomials $c \in P$, the message space $\mathscr{M}$ and the fact that $m = \mathrm{NR}_{\mathscr{G}}(c)$

where $\mathscr{G} = (g_1, \ldots, g_r)$ is secret. Moreover, because of the uniqueness of $\mathrm{NR}_{\mathscr{G}}(c)$, she does not need to find out actual polynomials $h_1, \ldots, h_s$ which are used by *Alice* for encrypting the plaintext message m. In fact, any other choice of polynomials $h'_1, \ldots, h'_s \in P$ for which $c = m + \sum h'_i p_i$ holds is equally fine for her. Therefore, she can think for the representation $c = \mathrm{NR}_G(c) + \sum h'_i p_i$ and hence has to estimate the maximal degree

$$d = \max\{\deg(h'_i p_i) \mid i = 1, \ldots, s\}.$$

This estimation could be d_c, the degree of the ciphertext polynomial c. If there is no cancellation in the top part of the sum $\sum h_i p_i$ then this will be the right estimation otherwise, d will be some number greater than d_c. We now summarize the method of this attack as follows:

Attack 3.8.1. *The Partial Gröbner Basis Attack*

Given an instance of CGBC, with public polynomials $p_1, \ldots, p_s$ and the ciphertext polynomial $c \in P$. Let J be the ideal generated by $\{p_1, \ldots, p_s\}$ and let the term ordering σ be degree compatible. Then for the partial Gröbner basis attack, the attacker, *Eve* performs the following steps to reveal the corresponding plaintext message $m \in \mathscr{M}$.

(1) Estimate the maximal degree $d = \max\{\deg(h'_i p_i)\}$ of the summands in a representation $c = \mathrm{NR}_{\sigma,\mathscr{G}}(c) + \sum_{i=1}^{s} h'_i p_i$ for which $\deg(h'_i p_i) \leq \deg(c)$ holds.

(2) Run the Buchberger Algorithm on $\{p_1, \ldots, p_r\}$ modified such that all operations involving polynomials of degree larger than d are not performed. The output will be a *partial* Gröbner basis $\mathscr{H}$ of the ideal J.

(3) Using the Division Algorithm, compute the normal remainder, $r = \mathrm{NR}_{\sigma,\mathscr{H}}(c)$. If $r \in \mathscr{M}$ then r is the required plaintext message m. Otherwise, increase d by one and repeat steps (2) and (3).

In step (1) of the above attack, the representation

$$c = \mathrm{NR}_{\sigma,G}(c) + \sum_{i=1}^{s} h'_i p_i \text{ for which } \deg(h'_i p_i) \leq \deg(c)$$

always exist (see [27] Proposition 2.1.1). Further, in the commutative setting, for the element c in the polynomial ideals, most of the times, it is not so difficult to generate enough Gröbner basis elements for the desired representation of c to exist. Therefore, theoretically, this attack seems to be very serious for the security of CGBC. In [8], where this attack was introduced, it has been described theoretically with the assumption that 'the polynomials in the public key are *low-degee dense* polynomials'. No experimental data is given to realize the effectiveness and the success of this attack when applied to some concrete cases. How the attack will work when these polynomials are not dense or when the degree bound for computing a partial Gröbner basis is very large? Is it always feasible to compute a partial Gröbner basis for a degree bound that is necessary for the success of this attack? In order to answer such questions and to examine the effectiveness and the success of this attack against a concrete instance of CGBC, it would be helpful to have a concrete public key and some ciphertexts available. Later, in Chapter 5, we shall examine the feasibility of this kind of attack when applied to some concrete instances of our proposed cryptosystem.

3.9 Chosen Ciphertext Attack and CGBC

The chosen ciphertext attack of Section 3.4 also applies to the case of CGBC. As explained earlier, to use the attack and to to break the cryptosystem, an attacker should have temporary access to the decryption algorithm. That is the attacker, *Eve* should be able to decrypt a limited number of "fake" ciphertext messages that she sends, without actually knowing *Bob*'s secret key. The attack in the setting that we are going to describe here was originally introduced by Bulygin [10] for attacking Rai's non-commutative Polly Cracker cryptosystem [41] but it also applies to CGBC. It is based on the fact that, given an ideal I of a polynomial ring $P = K[x_1, \ldots, x_n]$ and a term ordering σ on $\mathbb{T}^n$, if $\mathscr{G} = (g_1, \ldots, g_r)$ is a σ-Gröbner basis of I then we always have

$$\mathrm{NR}_{\sigma,\mathscr{G}}(\mathrm{LT}_\sigma(g_i)) = \mathrm{LT}_\sigma(g_i) - g_i.$$

Further, we assume that *Eve* knows or able to guess the leading terms of the secret polynomials in $\mathscr{G} = (g_1, \ldots, g_r)$. She then setup some "fake" ciphertext messages

c_i' of the form $\sum_j h_{ij}' p_j + \mathrm{LT}_\sigma(g_i)$. Using her temporary access to the decryption algorithm, she can then reveal complete secret key by decrypting each c_i'. For the sake of completeness, below we describe this attack in the setting of CGBC.

Attack 3.9.1. *Chosen Ciphertext Attack For CGBC*

Let $P = K[x_1, \ldots, x_n]$ be a polynomial ring and let σ be a term ordering on $\mathbb{T}^n$. Consider an instance of a CGBC with the secret key $G = \{g_1, \ldots, g_r\}$ and the public key $Q = \{p_1, \ldots, p_s\}$. For $i = 1, \ldots, r$, let $g_i = t_i + h_i$, with $t_i = \mathrm{LT}_\sigma(g_i)$ and note that t_i does not divide any monomial in h_i. Suppose that the attacker, *Eve* knows or can guess these leading terms of the polynomials $g_1, \ldots, g_r \in G$ and that she has temporary access to the decryption black box and can decrypt finite number of encrypted messages of her choice. Now she can recover the original secret key G by using the *chosen ciphertext attack* as follows:

For each $i = 1, \ldots, r$, she prepares "fake" ciphertext messages c_i' of the form

$$c_i' = t_i + \sum_j h_{ij}' p_j$$

by randomly choosing the polynomials $h_{ij}' \in P$. Then the basic set up of CGBC can give *Bob* no idea on how he can distinguish this fake ciphertext from the original one, i.e. from $c = m + h_1 p_1 + \cdots + h_s p_s$.

Now by using her access to the decryption algorithm, she decrypts these fake ciphertext polynomials c_i'. For each $i = 1, \ldots, r$, we have $\mathrm{NR}_{\sigma,\mathscr{G}}(\sum_j h_{ij}' p_j) = 0$. As a result, for each i, she gets

$$\mathrm{NR}_{\sigma,\mathscr{G}}(c_i) = -h_i.$$

And then by recombining, she recovers $g_i = t_i + h_i$.

Note that the success of this attack completely reveals the *Bob*'s secret key and hence the attacker can then decrypt any ciphertext $c = m + h_1 p_1 + \cdots + h_s p_s$ to recover the plaintext message m. The attack in this form also remains valid for the general non-commutative Gröbner basis Cryptosystem presented in [1].

In [42], T. Rai and S. Bulygin have proposed certain countermeasures to defeat Attack 3.9.1. We will come to these countermeasure while discussing security issues of our proposed cryptosystems in Chapter 5. The idea is not to make the complete set $\mathscr{O}_\sigma(I) = P \setminus \mathrm{LT}_\sigma(I)$ public. That is, the message space, $\mathscr{M}$ should not equal to $\mathscr{O}_\sigma(I)$ rather it should be a small subset of $\mathscr{O}_\sigma(I)$. In this way, it will not be difficult for *Bob* to detect fake ciphertext polynomials c_i' by publishing a subset $\mathscr{M} \subset \mathscr{O}_\sigma(I)$ such that the set

$$(\mathscr{O}_\sigma(I) \setminus \mathscr{M}) \cap \mathrm{Supp}(g_i) \neq \emptyset \text{ for all } i = 1, \ldots, r.$$

Then modify the decryption algorithm to return an error message whenever

$$\mathrm{NR}_{\sigma,\mathscr{G}}(c) \notin \mathscr{M}.$$

For further details we refer to [42].

3.10 General Gröbner Basis Cryptosystems

The successful cryptanalysis of specific instances of the Polly Cracker encryption scheme has put the security of CGBC in a great doubt. Except for the linear algebra attack, most of the other attacks are known to work only in the special case, that is, Polly Cracker. We call it a special case in the sense that the secret key is a tuple $(a_1, \ldots, a_n) \in K^n$, where K is a finite field and decryption is achieved by evaluating the ciphertext at this tuple which is supposed to be common zero of the polynomials in the public key (see Section 3.2). No further concrete hard instances of CGBC have been investigated or presented to confirm the failure of such cryptosystems. This, motivates researchers in this area to investigate other algebraic structures for constructing Gröbner basis type cryptosystems that might be secure against standard attacks or to use different strategies for encryption to make these attacks impossible to work. Among these tries, most prominent are the following attempts:

- Le Van Ly's **Polly Two** (see [35], an invariant of Polly Cracker scheme with the advantage that the usual linear algebra attacks do not work. Sine the attacks based on linear algebra (see 3.5 and 3.6) appeared to be most serious attacks on both the Polly Cracker cryptosystem and CGBC, it seems that in Polly Two, the only choice left for the attacker, *Eve* is to compute a possibly hard Gröbner basis. In [34] some concrete hard instances of Polly Two are given that are assumed to be difficult to break but, unfortunately, these instances have been successfully broken by R. Steinwandt using a side channel attack [47].

- T. Rai's **Non-commutative Polly Cracker Cryptosystems**, where to prevent linear algebra attacks, it has been suggested to construct Gröbner basis cryptosystems based on two-sided ideals in non-commutative polynomial rings (see [41]). In its original setting, non-commutative Polly cracker cryptosystems are vulnerable to chosen ciphertext attacks described in Section 3.4.1. To defeat this attack, various countermeasures are suggested in [42]. Moreover, the explicit instances of this cryptosystem given in [41] are based on principal ideals of free non-commutative associative algebras. It has been argued that the Gröbner basis of such principal non-commutative ideals can be infinite but, it is easy to compute and describe, and that the principal ideals might allow the easy recovery of the secret key by using the 'factoring attack'.

- The **Gröbner Basis Cryptosystems** (GBC) introduced by Ackermann and Kreuzer. This is a most general class of Gröbner basis type cryptosystems. These cryptosystems are based on the theory of Gröbner basis of modules over general non-commutative rings.

Remark 3.10.1. The security of this general class of GBC is strongly based on the difficulty of computing Gröbner bases of modules over non-commutative rings (see [1]). In general, the computation of Gröbner basis is EXPSPACE-hard. The advantage of using modules instead of ideals of the ring is that one can encode hard combinatorial or number theoretic problems in the action of the terms on the canonical basis vectors. Following well known cryptosystems are contained as **special cases** in this general class of GBC:

- *RSA (Rivest-Shamir- Adelmann) cryptosystem*,

- *ElGamal cryptosystem*,

- *Polly Cracker* and *(commutative) GBC*,

- *Polly 2*,
- *Braid group cryptosystem* (see [3]), and
- *Rai's non-commutative Polly cracker cryptosystem.*

In [1] the security issues of general Gröbner bases cryptosystems are also addressed and it is claimed that GBC are secure against various known standard attacks described in Section 3.3.

Being described in a "general" setting, it is important to construct a "Special Class" of such GBC with specific hard instances. Here we will not describe the complete theory of Gröbner basis of modules over general non-commutative monoid rings nor we explain GBC in this general setting. Instead we refer to [1] for details and use the idea of GBC to propose a new cryptosystem. For the design and implementation of this special class of cryptosystems we shall use Weyl Algebras (see Chapter 2) as base rings. In the next chapter, we will describe these "Weyl Gröbner basis Cryptosystems (WGBC)".

Chapter 4

Weyl Gröbner Basis Cryptosystems

In this chapter we will introduce cryptosystems which are special cases of the following Cryptosystem 4.1.1. This class of new cryptosystems is adapted from the very general setting of Gröbner Basis Cryptosystems 3.10, by using the Weyl algebra as the base ring. We have described Weyl algebras and their basic properties in Chapter 2. This special class of general GBC will be called "Weyl Gröbner Basis Cryptosystem (WGBC)" and will be described in Section 4.1 of this chapter. In Section 4.2 , we will introduce procedures for WGBC key generation and in Section 4.3 , we describe explicit instructions for constructing concrete instances of WGBC.

4.1 The WGBC

Using the notation from Chapter 2, let K be a field, and consider the Weyl algebra $A_n = K[x_1,\dots,x_n,\partial_1,\dots,\partial_n]$ of index n over K. The set of all standard terms of A_n is given by the set

$$B_n = \{x^\alpha \partial^\beta \mid \alpha,\beta \in \mathbb{N}^n\}.$$

Let σ be a term ordering on B_n. Further recall that, given a set of Weyl polynomials $G = \{g_1,\dots,g_r\} \subset A_n \setminus \{0\}$, we can use the *left Division Algorithm* to find out a *normal remainder* $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ of any polynomial $f \in A_n$ with respect to the tuple $\mathscr{G} = (g_1,\dots,g_r)$ (see Algorithm 2.3.18 and Definition 2.3.20). Moreover, if G is a left σ-Gröbner basis of an ideal I, then every Weyl polynomial f has a unique normal remainder $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ (see Theorem 2.4.1), and that if $f \in I$ then $\mathrm{NR}_{\sigma,\mathscr{G}}(f) = 0$

(Theorem 2.4.1, Part (2)). With these ingredients, we are now ready to introduce the following class of cryptosystems.

Cryptosystem 4.1.1. Given a Weyl algebra A_n of index n over K, let I be a non-trivial left ideal of A_n and let $G = \{g_1, \ldots, g_r\}$ be its left σ-Gröbner basis. We set $\mathscr{G} = (g_1, \ldots, g_r)$ and $\mathscr{O}_\sigma(I) = B_n \setminus \{\mathrm{LT}_\sigma(f) \mid f \in I \setminus \{0\}\}$. Then a left **Weyl Gröbner basis cryptosystem (WGBC)** consists of the following data.

(1) **Public Key** A set Q of Weyl polynomials $\{p_1, \ldots, p_s\}$ contained in $I \setminus \{0\}$ and a subset $\mathscr{M}$ of $\mathscr{O}_\sigma(I)$ are known publicly.

(2) **Secret Key:** The left σ-Gröbner basis $G = \{g_1, \ldots, g_r\}$ of the ideal I and the set $\mathscr{O}_\sigma(I)$ are kept secret.

(3) **Message Space:** The message space is the K-vector subspace $\langle \mathscr{M} \rangle_K$ of A_n generated by $\mathscr{M} \subset \mathscr{O}_\sigma(I)$.

(4) **Ciphertext Space:** The ciphertext units are Weyl polynomials in A.

(5) **Encryption:** For encrypting a plaintext message $m \in \langle \mathscr{M} \rangle_K$, choose Weyl polynomials $\ell_1, \ldots, \ell_s$ and compute the standard form of

$$c = m + \ell_1 p_1 + \cdots + \ell_s p_s.$$

to get the ciphertext polynomial c.

(6) **Decryption:** Given a ciphertext unit $c \in A_n$, compute $\mathrm{NR}_{\sigma,\mathscr{G}}(c)$. If the result is contained in $\langle \mathscr{M} \rangle_K$, return it. Otherwise, return c.

Note here that, since G is a σ-Gröbner basis of the ideal I and the polynomials $p_1, \ldots, p_s$ are contained in I, it follows that for each $i = 1, \ldots, s$, we have $\mathrm{NR}_{\sigma,\mathscr{G}}(p_i) = 0$ (see Theorem 2.4.1.2). This implies that

$$\mathrm{NR}_{\sigma,\mathscr{G}}(m + \ell_1 p_1 + \cdots + \ell_s p_s) = m,$$

which in turn implies the correctness of this system.

***Note*.** From now onwards, we abbreviate a left Weyl Gröbner basis cryptosystem as WGBC if no confusion can arise.

The security of WGBC strongly depends on the difficulty of computing Gröbner bases in Weyl algebras. That is, if an attacker can compute G, he can break the cryptosystem. Together with the subset of $\mathscr{O}_\sigma(I)$ the attacker only knows the Weyl polynomials $\{p_1,\dots,p_s\}$ in the public key $Q \subset I$. Therefore, they have to be created in a way that hides all the information about the system of generators of I. The attacker might try to compute a left σ-Gröbner basis of the ideal $J = \langle Q \rangle$ generated by the set of polynomials in the public key. In fact, in the settings of Weyl algebra, we can make this task difficult by suitably constructing the public polynomials $\{p_1,\dots,p_s\}$ such that the Gröbner basis of the ideal $J = \langle p_1,\dots,p_s \rangle$ is hard to compute. To show the existence of such ideals in Weyl algebras, below we give three examples using Weyl algebras over a field of characteristic 7, 3, and 0.

***Note*.** Throughout the thesis whenever we write 'our computing machine', we mean a computer system with 24 GB of RAM, and having the processor AMD Dual Opteron 2.4 GHz. All computations are performed on this computing machine and therefore all the timings are given accordingly.

Example 4.1.2. Consider the Weyl Algebra $A_3 = \mathbb{F}_7[x_1,x_2,x_3,\partial_1,\partial_2,\partial_3]$ of index 3 over the finite field $\mathbb{F}_7$ of characteristic 7 and let σ=`DegRevLex`. Choose the following Weyl polynomials of A_3,

$$\begin{aligned} f_1 &= -\partial_1^3\partial_3^5\partial_2^5 + x_3^5, \\ f_2 &= -3x_3\partial_3^5\partial_2^5 + x_3\partial_1^3, \\ f_3 &= -2\partial_1^4\partial_3^5 - x_1\partial_3^7 + x_2^3\partial_2^5. \end{aligned}$$

Let $I = \langle f_1, f_2, f_3 \rangle$ be the left ideal of A_3 generated by $\{f_1,f_2,f_3\}$. Then the reduced left σ-Gröbner basis of I is the set $G = \{g_1,\dots,g_{35}\}$ consisting of the following 35 polynomials in standard form

$\{\partial_1^5,\quad x_3^5,\quad x_3\partial_1^4\partial_3 + 3\partial_1^4,\quad x_3^3\partial_1^3,\quad x_2x_3^2\partial_1^3\partial_2\partial_3 + 3x_2x_3\partial_1^3\partial_2 + 3x_3^2\partial_1^3\partial_3 + 2x_3\partial_1^3,$
$x_2^3\partial_2^5 - x_1\partial_3^7,\quad x_2\partial_1^4\partial_2^3 - 2\partial_1^4\partial_2^2,\quad x_1x_2^3\partial_1^4 - 3x_2^3x_3\partial_1^3\partial_3 - 3x_2^3\partial_1^3,\quad \partial_1^4\partial_3^5,$
$x_2^3x_3^2\partial_1^3\partial_3 + 3x_2^3x_3\partial_1^3,\quad \partial_1^3\partial_3^7,\quad x_3\partial_1^3\partial_2^5\partial_3 + 3\partial_1^3\partial_2^5,\quad \partial_3^{19},$
$x_2^2x_3\partial_1^3\partial_2^3\partial_3 - x_2\partial_1^2\partial_3^7 - 2x_1x_2\partial_1^4\partial_2^2 + x_2^2\partial_1^3\partial_2^3 - 3x_2x_3\partial_1^3\partial_2^2\partial_3 - x_1\partial_1^4\partial_2 - 3x_2\partial_1^3\partial_2^2 +$
$3x_3\partial_1^3\partial_2\partial_3 + 3\partial_1^3\partial_2,\quad x_2^2x_3\partial_1^3\partial_2^4 + 3x_3\partial_1^2\partial_3^7 - 3x_2x_3\partial_1^3\partial_2^3 - 3x_3\partial_1^3\partial_2^2,$
$x_2^2\partial_1^3\partial_2^5 - 3x_2x_3\partial_1^3\partial_2^4\partial_3 + 3\partial_1^2\partial_2\partial_3^7 - 3x_2\partial_1^3\partial_2^4 + 3x_3\partial_1^3\partial_2^3\partial_3 + 3\partial_1^3\partial_2^3,$
$x_1x_2^2x_3\partial_1^4\partial_2^2 - 2x_2^3x_3\partial_1^3\partial_2^3 + x_1x_2x_3\partial_1^4\partial_2 + 3x_2^2x_3\partial_1^3\partial_2^2 - 2x_1x_3\partial_1^4 - x_2x_3\partial_1^3\partial_2 + 2x_3\partial_1^3,$

$x_3\partial_1^2\partial_3^8+3\partial_1^2\partial_3^7, \quad x_3^3\partial_3^{10}+x_3^2\partial_3^9-3x_3\partial_3^8-3\partial_3^7, \quad x_2^2\partial_1^2\partial_2^5\partial_3^5,$
$x_2\partial_1^2\partial_2\partial_3^7-2x_2x_3\partial_1^3\partial_2^3\partial_3+2\partial_1^2\partial_3^7-2x_2\partial_1^3\partial_2^3-3x_3\partial_1^3\partial_2^2\partial_3-3\partial_1^3\partial_2^2,$
$x_2x_3\partial_1^2\partial_3^7+2x_1x_2x_3\partial_1^4\partial_2^2+2x_22x_3\partial_1^3\partial_2^3+x_1x_3\partial_1^4\partial_2+x_2x_3\partial_1^3\partial_2^2-x_3\partial_1^3\partial_2,$
$x_2^2\partial_1^2\partial_3^7+3x_1x_2^2\partial_1^4\partial_2^2-2x_2^2x_3\partial_1^3\partial_2^2\partial_3+2x_1x_2\partial_1^4\partial_2-2x_2^2\partial_1^3\partial_2^2+x_2x_3\partial_1^3\partial_2\partial_3-2x_1\partial_1^4+$
$x_2\partial_1^3\partial_2-x_3\partial_1^3\partial_3-\partial_1^3, \quad x_3\partial_2^5\partial_3^5+2x_3\partial_1^3, \quad x_3^3\partial_2^5\partial_3^3+x_3^2\partial_2^5\partial_3^2-3x_3\partial_2^5\partial_3-3\partial_2^5,$
$x_2^4x_3\partial_1^3\partial_2^3-x_1x_2^2x_3\partial_1^4\partial_2-2x_2^3x_3\partial_1^3\partial_2^2-3x_1x_2x_3\partial_1^4+x_2^2x_3\partial_1^3\partial_2+3x_2x_3\partial_1^3,$
$x_3^3\partial_1^2\partial_3^7, \quad x_2x_3\partial_1^3\partial_2\partial_3^6-x_2\partial_1^3\partial_2\partial_3^5+3x_3\partial_1^3\partial_3^6-3\partial_1^3\partial_3^5, \quad \partial_1\partial_3^{12}, \quad x_3\partial_3^{12}+2x_2^3x_3\partial_1^4,$
$x_2^3x_3\partial_1^3\partial_3^6-x_2^3\partial_1^3\partial_3^5, \quad \partial_1^3\partial_2^5\partial_3^5, \quad x_2\partial_1^2\partial_2^5\partial_3^6-3x_3\partial_1^2\partial_2^4\partial_3^7-2x_2x_3\partial_1^3\partial_2^7+3x_3\partial_1^3\partial_2^6,$
$x_2x_3^2\partial_1^3\partial_2^8-3x_2\partial_1^2\partial_2^6\partial_3^5+3x_3^2\partial_1^3\partial_2^7,$
$x_3^2\partial_1^2\partial_2^4\partial_3^7+3x_2x_3^2\partial_1^3\partial_2^7-2x_2\partial_1^2\partial_2^5\partial_3^5-x_3^2\partial_1^3\partial_2^6\}.$

From the polynomials $\{f_1, f_2, f_3\}$, let us now create polynomials p_1 and p_2 as follows:

$$p_1 = h_{11}f_1+h_{12}f_2+h_{13}f_3 \quad \text{and} \quad p_2 = h_{21}f_1+h_{22}f_2+h_{23}f_3$$

By choosing

$$\begin{array}{rclrclrcl} h_{11} & = & -2\partial_1+\partial_2^5\partial_3^5, & h_{12} & = & -2x_3^4, & h_{13} & = & \partial_2^5, \\ h_{21} & = & -2\partial_1+x_3, & h_{22} & = & \partial_1^3, & h_{23} & = & \partial_2^5, \end{array}$$

we then have,

$$\begin{array}{rcl} p_1 & = & -\partial_1^3\partial_2^{10}\partial_3^{10}+x_2^3\partial_2^{10}-3x_3^4\partial_2^5\partial_3^4-x_1\partial_2^5\partial_3^7+x_2^2\partial_2^9-3x_3^3\partial_2^5\partial_3^3-3x_2\partial_2^8- \\ & & 2x_3^2\partial_2^5\partial_3^2-2x_3^5\partial_1^3-3\partial_2^7-2x_3\partial_2^5\partial_3-2x_3^5\partial_1+\partial_2^5, \\ p_2 & = & 3x_3\partial_1^3\partial_2^5\partial_3^5+x_2^3\partial_2^{10}-x_1\partial_2^5\partial_3^7+x_2^2\partial_2^9-3x_2\partial_2^8+x_3\partial_1^6-3\partial_2^7+x_3^6-2x_3^5\partial_1. \end{array}$$

Let $J = \langle p_1, p_2\rangle$ be the left ideal generated by the polynomials p_1 and p_2. We claim that the Gröbner basis of the ideal J is very hard to compute using current resources and implementation of algorithms for the computation of Gröbner bases of ideals in Weyl algebras. We were unable to compute this Gröbner basis using the implementation of these algorithms on Singular, ApCoCoA, and Macaulay 2 on our computing machine.

Note. It has been observed that this claim remains valid if we change the characteristic p to 3 and 5 in the above Example 4.1.2. Moreover, the ideal I becomes a trivial ideal for characteristic $p \geq 13$. That is, for $p \geq 13$, we have $G = \{1\}$.

Below we give another example by considering the Weyl algebra A_3 over the prime field of characteristic 3.

Example 4.1.3. Consider the Weyl algebra $A_3 = \mathbb{F}_3[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$ of index 3 over the field $K = \mathbb{F}_3$, and let the monomial ordering on A be $\sigma =$ `DegRevLex`. Choose the following Weyl polynomials of A_3

$$f_1 = -\partial_1^3\partial_2^5\partial_3^5 + x_2^5, \quad f_2 = x_2\partial_3^5 + \partial_1^3, \text{ and } f_3 = \partial_1^4\partial_2^5 - x_1\partial_2^7.$$

Let $I = \langle f_1, f_2, f_3\rangle$ be the left ideal generated by f_1, f_2, and f_3. Then the reduced left σ-Gröbner basis of the ideal I is the set[1] G consisting of 26 Weyl polynomials in standard form.
Let us now construct an ideal $J = \langle p_1, p_2\rangle$, where

$$p_1 = h_{11}f_1 + h_{12}f_2 + h_{13}f_3 \text{ and } p_2 = h_{21}f_1 + h_{22}f_2 + h_{23}f_3$$

and where we let

$$\begin{array}{rclrclrcl} h_{11} &=& x_2 + \partial_1, & h_{12} &=& \partial_2^4\partial_3^5 + \partial_1^3\partial_2^5, & h_{13} &=& \partial_3^5 - \partial_1^2, \\ h_{21} &=& \partial_1\partial_2\partial_3, & h_{22} &=& -\partial_1^4\partial_2^5, & h_{23} &=& \partial_2\partial_3^6 + x_2\partial_3^5. \end{array}$$

We get

$$\begin{array}{rcl} p_1 &=& x_2\partial_2^4\partial_3^{10} - x_1\partial_2^7\partial_3^5 + \partial_2^3\partial_3^{10} + x_1\partial_1^2\partial_2^7 - \partial_1\partial_2^7 + x_2^6 + x_2^5\partial_1, \\ p_2 &=& -x_1\partial_2^8\partial_3^6 - x_1x_2\partial_2^7\partial_3^5 + \partial_1^4\partial_2^4\partial_3^5 - \partial_1^7\partial_2^5 + x_2^5\partial_1\partial_2\partial_3 - x_2^4\partial_1\partial_3. \end{array}$$

Again, based on experiments carried out on our computing machine, we claim that the Gröbner basis of the ideal J is hard to compute. For instance, the implementation of the Buchberger Algorithm 2.3.24 on `Macaulay2` took 7,924 minutes of CPU time on our computing machine after which we interrupted the process to terminate without an output. At the time of interruption, we still had untreated 1777 S-polynomials with a total number of $59,196,454$ monomials.

In the above Examples 4.1.2 and 4.1.3 we have considered the Weyl Algebra A over a field of positive characteristic. We have seen in this case that given a non-trivial left ideal $I \subset A$, it is possible to construct an ideal $J \subset I$ such that Gröbner basis of the ideal J is hard to compute. Our next example shows that such ideals can also be created for Weyl algebras over a field of characteristic zero.

[1]This set G is given in Appendix C.2.

Example 4.1.4. Consider the Weyl algebra $A_3 = \mathbb{Q}_3[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$ of index 3 over the field $\mathbb{Q}$ of characteristic 0, and let the term ordering be $\sigma = \texttt{DegRevLex}$. Choose following Weyl polynomials

$$f_1 = 2x_2^3\partial_2^3 + 3x_1^3\partial_1^2, \quad f_2 = -x_2^2\partial_3^5 + \partial_1^3, \text{ and } \; f_3 = x_3^3\partial_3^3 - x_1^2\partial_1^3.$$

Le $I = \langle f_1, f_2, f_3\rangle$ be the left ideal generated by these polynomials. Then a left σ-Gröbner basis of the ideal I is given by the set

$$G = \{\partial_1^2, \partial_3^3, x_2^3\partial_2^3\}.$$

Let us now construct an ideal $J = \langle p_1, p_2\rangle$, where

$$p_1 = h_{11}f_1 + h_{12}f_2 + h_{13}f_3 \;\text{ and }\; p_2 = h_{21}f_1 + h_{22}f_2 + h_{23}f_3$$

and where we let

$$\begin{aligned} h_{11} &= \partial_3^5, & h_{12} &= x_3^2 + 2x_2\partial_2^3, & h_{13} &= x_2^2\partial_3^2, \\ h_{21} &= x_3^3\partial_3^3 + \partial_2\partial_3^5, & h_{22} &= 6\partial_2^3, & h_{23} &= -2x_2^3\partial_2^3. \end{aligned}$$

We get

$$\begin{aligned} p_1 &= -x_2^2x_3^3\partial_3^5 + 3x_1^3\partial_1^2\partial_3^5 - 12x_2^2\partial_2^2\partial_3^5 + x_2^4\partial_3^4 + 2x_2\partial_1^3\partial_2^3 - 12x_2\partial_2\partial_3^5 + x_3^3\partial_1^3, \\ p_2 &= 2x_2^3\partial_2^4\partial_3^5 + 2x_1^2x_2^3\partial_1^3\partial_2^3 + 3x_1^3x_3^3\partial_1^2\partial_3^3 + 3x_1^3\partial_1^2\partial_2\partial_3^5 - 36x_2\partial_2^2\partial_3^5 + 6\partial_1^3\partial_2^3 - \\ &\quad 36\partial_2\partial_3^5 \end{aligned}$$

With these settings, the left Gröbner basis of the ideal $J = \langle p_1, p_2\rangle$ turned out to be very hard to compute. In this case we fail to compute the Gröbner basis due to very fast growth of memory required for the computations. For instance, using the CAS `Macaulay2` on our computing machine, we terminated the process of Gröbner basis computation of the ideal J after 4004 minutes of CPU time. At the time of interruption, the intermediate results had grown enough to consume 18.7 GB of system memory. The same computation also fails to complete on the computer algebra systems ApCoCoA and `Singular`.

Note. In Examples 4.1.2, 4.1.3, and 4.1.4, the intermediate results during computation show that, the computation of Gröbner bases of carefully constructed ideals in Weyl algebras fails to complete because of the following reasons:

- the memory required to store the intermediate results grows too fast,
- due to the increase in the size of the polynomials during the computation, the reduction process (Division Algorithm 2.3.18) gets very slow. That is, reduction of S-polynomials slows down as the computation grows.

Now an obvious question arises: '*Can we use such ideals for the construction of practical concrete instances of WGBC*?' As explained in Chapter 3, successful cryptanalysis of Gröbner Basis Cryptosystems might be possible by using certain attacks where the attacker does not need to compute the Gröbner basis of the ideal $J \subset I$. For example, the *chosen ciphertext attack* and the attacks based on *linear algebra* can be applied. Moreover, instead of computing a complete Gröbner basis of the ideal J, the attacker can also try using partial Gröbner bases of J for the *partial Gröbner basis attack*. Therefore, choosing an ideal $J \subset I$ such that Gröbner basis of the ideal J is hard to compute is not sufficient for constructing a secure instance of a WGBC. Together with this condition, we also have to make sure that, on a particular instance of WGBC that we construct, the above standard attacks cannot be successful to break the system. To achieve this goal, we have to fix certain parameters of the WGBC and the way of constructing polynomials $p_1, \dots, p_s \in I$ for the public key Q.

Notation 4.1.5. Parameters of a WGBC: In order to make Cryptosystem 4.1.1 usable, we have to quantify certain parameters for the key generation and the way of choosing the polynomials $\ell_1, \dots, \ell_s$ for the encryption process. Here are the various parameters that we have to consider for constructing an instance of WGBC that might be hard to break:

* p: the characteristic of the base field K,
* n: the index of the Weyl algebra A,
* σ: the term ordering on A,
* s: the size of the public key Q,
* d_g: the maximum degree of the polynomials $g_1, \dots, g_r$ in the secret key G,
* d_p: the maximum degree of the polynomials $p_1, \dots, p_s \in Q$,
* d_ℓ: the maximum degree of the polynomials $\ell_1, \dots, \ell_s$ used for the encryption.

The efficiency and the security of Weyl Gröbner basis cryptosystems greatly depends on the right choice for these parameters. For example, we shall see in the coming sections that the degree d_ℓ and number of terms in the polynomials $\ell_1, \dots, \ell_s$ can make the size of the resulting ciphertexts too large and result in a bad data-rate for transmissions. The large size of the ciphertext might also decrease the efficiency by increasing the time taken by the decryption process. Moreover, we need to specify the values for the degree d_ℓ for a guaranteed security against *partial Gröbner basis attacks*.

In the next section, we describe in detail the key generation and implementation in order to construct a practical instance of a WGBC. This parameter consideration is also important to defeat the attacks based on linear algebra.

4.2 WGBC Key Generation and Implementation

The aim of this section is to introduce a step-by-step procedure for generating a pair (G,Q) for constructing a secure instance of WGBC. Keeping in mind the observations and the experimental results from the examples of the last section, we introduce following procedure for the way of creating a secure secret key and a presumably hard to break ciphertext c.

Procedure 4.2.1. In the above setting of Cryptosystem 4.1.1 perform the following steps.

(1) Choose a set of Weyl polynomials $G = \{g_1, \dots, g_r\}$ which form a reduced left σ-Gröbner basis of the left ideal $I = \langle G \rangle \subset A_n$.

(2) For $i = 1, \dots, s$ and $j = 1, \dots, r$, choose the polynomials $h_{ij} \in A_n$ and compute the standard form of the Weyl polynomials

$$p_i = h_{i1}\, g_1 + \cdots + h_{ir}\, g_r.$$

While choosing the polynomials h_{ij}, make sure that following properties hold.

   (a) The degree forms $\mathrm{DF}(h_{ij} g_j)$ of highest degree cancel. The other degree forms $\mathrm{DF}(h_{ij} g_j)$ cancel or their coefficients are changed in p_i by the process of converting the remaining $h_{ik} g_k$ to standard form.

(b) There are sufficiently high powers of $\partial_1, \dots, \partial_n$ in the terms of the support of h_{ij} such that, after bringing $h_{ij}g_j$ to standard form, no information about $\mathrm{Supp}(g_j)$ is leaked in $\mathrm{Supp}(p_i)$. In particular, the leading terms $\mathrm{LT}_\sigma(g_1), \dots, \mathrm{LT}_\sigma(g_r)$ should be well hidden.

(c) In verifying properties (a) and (b) above, make sure that there are no gaps in the degrees of various terms in $\mathrm{Supp}(p_i)$. That is, for each i, if $\deg(p_i) = d_{p_i}$, then $\mathrm{Supp}(p_i)$ should contain a sufficient number of terms of each degree between d_{p_i} and 1. In fact, this reduces the sparsity of the polynomials $p_1, \dots, p_s$.

(3) Let $J = \langle p_1, \dots, p_s \rangle$ be the left ideal generated by the polynomials in the public key Q. Make sure that not only the complete left σ-Gröbner basis of the ideal J is hard to compute, but also *partial Gröbner bases* are infeasible to compute for large degree bounds.

(4) Choose a small enough subset $\mathscr{M} \subset \mathscr{O}_\sigma(I)$ for the message space $\langle \mathscr{M} \rangle_K$ in such a way that every g_i contains at least one term in $\mathscr{O}_\sigma(I) \setminus \mathscr{M}$.

(5) For constructing a ciphertext polynomial

$$c = \ell_1 p_1 + \cdots + \ell_s p_s + m,$$

choose the polynomials $\ell_1, \dots, \ell_s$ such that the following properties hold:

(a) Make sure that $\mathrm{Supp}(\ell_1 p_1 + \cdots + \ell_s p_s)$ contains all terms of $\mathrm{Supp}(m)$ and many terms of $\mathscr{M}$. In this way, the monomials of m will be either cancelled or their coefficients will be changed in the lower-degree part of the polynomial c.

(b) Ascertain that the degree forms $\mathrm{DF}(\ell_i p_i)$ of highest degree cancel in c, and that the other degree forms $\mathrm{DF}(\ell_i p_i)$ cancel or their coefficients are changed in c by the process of converting the remaining $\ell_i p_i$ to standard form.

(c) Again, in meeting properties (a) and (b) above, use sufficiently high powers of $\partial_1, \dots \partial_n$ in the terms of the support of ℓ_i such that, after bringing $\ell_i p_i$ to standard form, there are no wide gaps in degrees of various terms in $\mathrm{Supp}(c)$. This means that the sparsity of the ciphertext polynomial will be reduced.

(6) Make sure that the above choices of the polynomials $\ell_1, \dots, \ell_s$ make the degree, d_c, of the ciphertext c high enough such that no partial Gröbner basis of the ideal J can be computed up to the degree bound d_c. Moreover, if $\mathscr{H}$ is a partial Gröbner basis of J up to a degree bound $d < d_c$, then $\mathrm{NR}_{\sigma,\mathscr{H}}(c) \neq m$.

Remark 4.2.2. We have seen in Chapter 2, that due to the structure of Weyl multiplication, the product of Weyl polynomials in standard form blows up to include many terms. In fact, in the Weyl algebra A_n, we can have $t, t' \in B_n$ such that the product tt' is a polynomial having many terms in its standard form. Therefore, by including powers of $\partial_1, \dots, \partial_n$ in the polynomials $\ell_1, \dots, \ell_s$, we can make the lower and the middle part of the ciphertext polynomial $c = \ell_1 p_1 + \cdots + \ell_s p_s + m$ dense enough to hide the message m, and to accomplish the steps (5) and (6) of Procedure 4.2.1. With the same strategy, we can fulfil the above requirement (2).(b) of Procedure 4.2.1.

In the next chapter, we shall explain why we believe that, by completing the steps of Procedure 4.2.1, we can make the standard attacks infeasible. In fact, step (2) makes sure that the polynomials in the secret key G are well concealed. The step (5) ensures that not only the plaintext message m is well hidden in the ciphertext polynomial c, but by reducing the sparsity of the polynomial c and removing gaps in the degrees of the terms in the support of c we are also, making linear algebra attacks harder to apply. Similarly, by completing the steps (3) and (4), we are, respectively, making the *chosen ciphertext attack* and the *partial Gröbner basis attack* infeasible.

Remark 4.2.3. In Step (4) of Procedure 4.2.1, we have suggested to use the reduced σ-Gröbner bases of the ideal I considered for constructing an instance of a WGBC. By definition of WGBC in Cryptosystem 4.1.1, *Bob*, can take any left σ-Gröbner basis of I for such construction, but, for all our experimental results and instances of WGBC that will be presented in this thesis, most of the time we will be using the reduced Gröbner bases unless otherwise specified.

4.3 Construction of Hard Instances

For constructing concrete hard instances of Weyl Gröbner Bases Cryptosystems, the structure of Weyl algebras is very useful in satisfying the requirements of Procedure 4.2.1. In the next procedure, we shall provide an explicit suggestion how this can be done. The idea is based on Proposition 2.5.2 for constructing non-trivial left ideals of A_n (see Example 2.5.3).

Procedure 4.3.1. Let $K = \mathbb{F}_p$ be a finite field of characteristic p, let $n \geq 2$, and consider the Weyl algebra A_n of index n over K. Let σ be a term ordering on B_n. Then the following instructions define a WGBC which satisfies Conditions (1) – (6) of Procedure 4.2.1.

(1) For $i = 1, \ldots, r$, with $2 \leq r \leq n$, choose a (random) polynomial $g_i \in K[x_i, \partial_i] \subseteq A_n$ such that:
   (a) $\deg(g_i) \geq d'$,
   (b) the number of terms in support of each g_i is at least N.

   Let $G = \{g_1, \ldots, g_r\}$ be the set of these polynomials, and let $I = \langle G \rangle$ be the left ideal generated by G. By Proposition 2.5.2, the set G is a left σ-Gröbner basis of I.

(2) For the message space, choose the set $\mathscr{M} \subseteq \mathscr{O}_\sigma(I)$ such that every g_i has at least one term from $\mathscr{O}_\sigma(I) \setminus \mathscr{M}$ in its support.

(3) Now create Weyl polynomials $p_1, \ldots, p_s$ of the form $p_i = h_{i1}g_1 + \cdots + h_{ir}g_r$ such that Conditions (2a)– (2c) of Procedure 4.2.1 are satisfied. In particular, choose the degree forms of the polynomials $h_{i1}, \ldots, h_{ir}$ such that they are a syzygy of $\mathrm{DF}(G)$, at least in the top-degree.

Remark 4.3.2. In Step (1) of the above procedure, the lower bounds D and N are suggested, respectively, for the degree and number of terms in the support of each g_i. Based on our experimentations and computations, it turns out that $D = 10$ and $N = 5$ are good choices for meeting the requirements of Procedure 4.2.1.

In the next chapter we shall see that if we establish an instance of WGBC by following the instructions in the above procedure, then such a system will be secure against standard known attacks.

Let us now use the instructions of this procedure to formulate a concrete case of WGBC.

Example 4.3.3. Let us take the Weyl algebra

$$A_2 = K[x_1, x_2, \partial_1, \partial_2]$$

over the finite field $K = \mathbb{F}_{13}$ of characteristic 13. Let the term ordering on the set B_2 of all terms of A_2 be `DegRevLex`. With these ingredients, we introduce the following WGBC:

(1) **Secret Key:**
Let $G = \{g_1, g_2\}$ be given by

$$\begin{aligned} g_1 &= 7x_1^7\partial_1^7 + 2x_1^6\partial_1^6 + 4x_1^2\partial_1^2 + 3x_1^3 - \partial_1^3 + x_1^2 - 3x_1\partial_1 - 2\partial_1^2 + 5x_1 - 7\partial_1 + 1 \\ g_2 &= 4x_2^5\partial_2^5 + 3x_2^4\partial_2^4 + 5x_2^4 + \partial_2^4 - 3x_2^3 - 4\partial_2^3 + x_2^2 - x_2\partial_2 + 2\partial_2^2 - 3 \end{aligned}$$

and let $I = \langle g_1, g_2 \rangle$ be the left ideal generated by G. The secret key is now the set G and let $\mathscr{G} = (g_1, g_2)$.

(2) **Public Key:**
Compute the standard form of the Weyl polynomials

$$p_1 = h_{11}\, g_1 + h_{12}\, g_2 \;\text{ and }\; p_2 = h_{21}\, g_1 + h_{22}\, g_2,$$

where

$$\begin{aligned} h_{11} &= 4x_1^3x_2^{11}\partial_1^3\partial_2^9 + 5x_1^3x_2^{10}\partial_1^3\partial_2^8 + 2x_1x_2^5\partial_2^5 + 2x_2^5\partial_1\partial_2^5 + 5x_1 - 3x_2 + \\ &\quad 2\partial_1 - 6\partial_2 + 3, \\ h_{12} &= 6x_1^{10}x_2^6\partial_1^{10}\partial_2^4 - 6x_1^9x_2^6\partial_1^9\partial_2^4 - 4x_1^8\partial_1^7 + 3x_1^7\partial_1^8 + 4x_2 + 2\partial_2 + 4, \\ h_{21} &= 5x_1^2x_2^{14}\partial_1^6\partial_2^{16} - 4x_1^2x_2^{13}\partial_1^6\partial_2^{15} - 7x_1 + 2x_2 + 4, \\ h_{22} &= x_1^9x_2^9\partial_1^{13}\partial_2^{11} + 7x_1^8x_2^9\partial_1^{12}\partial_2^{11} + 6\partial_1 - 3\partial_2 + 1. \end{aligned}$$

Then the Weyl polynomial p_1 has degree 36 and its standard form consists of 170 terms. Note here that, as suggested in Part (3) of Procedure 4.3.1, our choice of the polynomials h_{11} and h_{12} is such that the degree form $\mathrm{DF}(h_{11}g_1)$

and $\mathrm{DF}(h_{12}g_2)$ cancel in p_1 and many other terms are cancelled or their coefficients are changed in p_1. For instance, since $\mathrm{LM}_\sigma(g_1) = -6x_1^7\partial_1^7$, we choose a random term $t_1 = 4x_1^3x_2^{11}\partial_1^3\partial_2^9$ of degree 26 for h_{11}. Now the leading monomial of the product $t_1\,g_1$ is $2x_1^{10}x_2^{11}\partial_1^{10}\partial_2^9$ and to cancel it in p_1, we choose the monomial $t_1' = 6x_1^{10}x_2^6\partial_1^{10}\partial_2^4$ for h_{12}. If required, we proceed the same way for cancelling the terms in $\mathrm{DF}(t\,g_1)$. Note that we have $\mathrm{DF}(t_1'\,g_2) = -2x_1^{10}x_2^{11}\partial_1^{10}\partial_2^9$ and it will not appear in p_1. In order to make lower part of p_1 dense enough, we make use of Weyl multiplication by inserting lower-degree terms both in h_{11} and h_{12}. For instance, we choose a monomial $t_2 = 5x_1$ for h_{11} and to cancel the leading term of the product $t_2\,g_1$ we insert the monomial $t_2' = 3x_1^8\partial_1^7$ in h_{12} and again for the cancellation insert $t_3 = 2x_1x_2^5\partial_2^5$ in h_{11}. Continuing this way, we keep on adding and setting various terms for h_{11}, and h_{12} and finally compute p_1 as above. The degree of p_1 is 36 which means that all the terms of degree greater than 36 are cancelled in p_1. In this way, many terms in p_1 are either cancelled or their coefficients are changed. This can be easily seen by observing the number of terms in the homogeneous components of $h_{11}\,g_1$, $h_{12}\,g_2$, and $h_{13}\,g_4$ and comparing them with the number of terms of the homogeneous components of p_1, for instance, by using a CAS. Similarly, to compute p_2 we choose the above polynomials h_{21} and h_{22}. The Weyl polynomial p_2 has degree 48 and there are 128 terms in its standard form. Again, the highest degree terms cancel in p_2. The set $Q = \{p_1, p_2\}$ is now our public key[2].

(3) **Message Space:**
For the message space, we choose the K-vector space generated by the set

$$\mathscr{M} = \{x^\alpha\partial^\beta \mid |\alpha| \leq 11,\ |\beta| \leq 7\}.$$

There are 13^{2808} different possible plaintext units. Moreover, both secret polynomials g_1 and g_2 have terms from $\mathscr{O}_\sigma(I) \setminus \mathscr{M}$ in their support.

(4) **Encryption:**
To encrypt a message $m \in \langle\mathscr{M}\rangle_K$, we choose sparse polynomials ℓ_1, ℓ_2 of suf-

[2]These polynomials p_1 and p_2 are given in Appendix C.2.(2).

ficiently high degree and compute the standard form of the ciphertext polynomial

$$c = m + \ell_1 p_1 + \ell_2 p_2.$$

For instance, let us encrypt

$$m = -6x_2^4\partial_2^3 + 6\partial_2^6 + 5x_2^4 - \partial_2^4 + 6x_2^3 + 6\partial_2^3 + x_1^2 + x_2\partial_2 - 3\partial_1\partial_2 + 2x_1 + 2\partial_1 - 5$$

By choosing

$$\begin{aligned} \ell_1 &= -5x_1^{10}x_2^{16}\partial_1^{12}\partial_2^{19} - 2x_1^8x_2^{18}\partial_1^{10}\partial_2^{21} - \partial_1 + 1, \\ \ell_2 &= 4x_1^{11}x_2^{13}\partial_1^9\partial_2^{12} - 6x_1^9x_2^{15}\partial_1^7\partial_2^{14} + 2\partial_2 + x_2 + 2, \end{aligned}$$

in the above representation of c, we obtain the ciphertext polynomial c of degree 91 and its standard form consists of 2954 terms. Considering the size of the message space, the message expansion is rather moderate. The polynomials ℓ_1, ℓ_2 are chosen such that the degree forms of $\ell_1 p_1$ and $\ell_2 p_2$ are cancelled in c and to make the degree d_c high enough to meet the requirement (6) in Procedure 4.2.1. This can be achieved for instance in the same way as described the way of choosing h_{11} and h_{12} in the above key generation process. Moreover, lower-degree terms in ℓ_1, ℓ_2 are selected to make the message m well-hidden.

(5) **Decryption:**
Since $m = \mathrm{NR}_{\sigma,\mathscr{G}}(c)$, therefore to decipher c, it suffices to compute the normal remainder of the ciphertext polynomial c with respect to the secret key $\mathscr{G}$. In the present case, an efficient implementation of the Division Algorithm 2.3.18, recovers m in a couple of seconds.

The reason why we had to go up to rather high degrees in this example is clearly the fact that we used the Weyl algebra of index 2. As soon as we add a few more indeterminates, i.e. for $n > 2$, we gain additional freedom for the message space and the usual attacks on the Gröbner basis type cryptosystems become more difficult to carry out.

Note. For the instance of WGBC given in Example 4.3.3, observe that $\mathrm{Supp}(c) \setminus \mathrm{Supp}(\ell_1 p_1 + \ell_2 p_2)$ contains only 2 terms. This indicates that the message m is well hidden in the ciphertext c. Moreover, the number of terms in the homogeneous components of the ciphertext c are distributed as follows

$$\{(91,7),(90,2),(89,17),(88,8),(87,26),(86,11),(85,31),(84,20),(83,38),\\ (82,30),(81,42),(80,43),(79,55),(78,52),(77,61),(76,60),(75,71),(74,79),\\ (73,78),(72,92),(71,88),(70,94),(69,94),(68,96),(67,87),(66,92),(65,84),\\ (64,84),(63,72),(62,84),(61,82),(60,81),(59,75),(58,63),(57,57),(56,47),\\ (55,39),(54,28),(53,18),(52,12),(51,6),(50,16),(49,22),(48,33),(47,39),\\ (46,30),(45,36),(44,25),(43,28),(42,20),(41,24),(40,19),(39,22),(38,19),\\ (37,17),(36,13),(35,12),(34,11),(33,12),(32,11),(31,13),(30,13),(29,14),\\ (28,14),(27,15),(26,13),(25,12),(24,11),(23,10),(22,7),(21,3),(20,6),\\ (19,8),(18,14),(17,10),(16,14),(15,9),(14,13),(13,8),(12,11),(11,5),(10,5),\\ (9,3),(8,2),(7,2),(6,16),(5,28),(4,27),(3,19),(2,9),(1,4),(0,1)\}$$

where the tuple (n_1,n_2) indicates that the total number of terms of degree n_1 is n_2. This shows that the highest degree terms are cancelled in c and that the ciphertext contains many terms from the message space $\langle \mathscr{M} \rangle_K$.

In the next chapter, we shall come back to this instance of WGBC for further investigations and discuss the resistance of this system with respect to several standard attacks.

Our next procedure for the key generation of WGBC is based on the idea of using a randomly chosen left ideal of a Weyl algebra A_n. That is, we choose an ideal of A_n whose generators are selected as random Weyl polynomials. In order to proceed this way one has to be extra careful in choosing generating polynomials of such an ideal $I \subset A$ (see Section 2.5 for details). The selection of these polynomials is not purely random as we have to make sure that the ideal generated should have a non-trivial Gröbner basis.

Remark 4.3.4. Recall from Section 2.5, choosing a non-trivial ideal $I = \langle f_1, \ldots, f_q \rangle$ of Weyl algebras is not a trivial task when generating polynomials $f_1, \ldots, f_q$ are randomly chosen elements of A_n. From our experiments and computations, we have observed that higher the degree and number of terms in $\mathrm{Supp}(f_i)$, more time consuming and difficult will be the computation of a left σ-Gröbner basis of I (see also note at the end of Example 4.3.6). On the other hand, we also do not want the left Gröbner basis G of such ideals contains very few elements or, is very easy to guess from the public information. After all, we are interested in those ideals, such that a practical instance of WGBC can be built on them by meeting the requirements of Procedure 4.2.1. Therefore, based on our computational results, in order to choose a polynomial $f_i \in A_n$ for the generating system of a left ideal I, we suggest to choose f_i such that $\deg(f_i) \geq 6$ and the number of terms in $\mathrm{Supp}(f_i)$ is at least 3. It will be very likely that the ideal constructed this way will be a non-trivial ideal of A_n and that it can be used to make a practical instance of a WGBC satisfying requirements of Procedure 4.2.1. We will use these suggestions in the following procedure.

Procedure 4.3.5. Consider the Weyl algebra $A_n = K[x_1,\dots,x_n, \partial_1,\dots,\partial_n]$ of index n over a prime[3] field $K = \mathbb{F}_p$ of characteristic p. Let $n \geq 3$ and let σ be a term ordering on B_n. Then the following instructions define a WGBC which satisfies the requirements of Procedure 4.2.1.

(1) For $i = 1,\dots,u$, choose a random Weyl polynomial $f_i \in A_n \setminus K$, such that $\deg(f_i) \geq 6$ and $\#\mathrm{Supp}(f_i) \geq 3$. Moreover, these polynomials should be such that the left ideal $I = \langle f_1,\dots,f_u\rangle$ is a non-trivial ideal of A_n. Let the set $G = \{g_1,\dots,g_r\}$ be a left σ-Gröbner basis of the ideal I. Make sure that the size of this secret key is at least 8, i.e. $r \geq 8$.

(2) For the message space, choose the set $\mathscr{M} \subseteq \mathscr{O}_\sigma(I)$ such that at least 80 percent of polynomials in G are such that they have at least one term from $\mathscr{O}_\sigma(I) \setminus \mathscr{M}$ in their supports.

(3) For $i = 1,\dots,s$, create Weyl polynomials p_i of the form

$$p_i = h_{i1}\, g_1 + \cdots + h_{ir}\, g_r$$

where $h_{ij} \in A_n$ are chosen such that the computation of a left σ-Gröbner basis of the ideal $J = \langle p_1,\dots,p_s\rangle$ is infeasible and such that Conditions (2a)– (2c)

[3] We have also performed experiments with $K = \mathbb{Q}$, it turns out that in this case, firstly, choosing a random non-trivial ideal of our interest is rather involved task, and secondly, it is very difficult to control the sizes of polynomials in Q and growth of the ciphertext c.

of Procedure 4.2.1 are satisfied. Make sure that if some $g_j \in G$ does not satisfy Condition (2), then set the corresponding $h_{ij} = 0$ in the above representation of p_i. That is, use a $g_j \in G$ for the construction of polynomials $p_1, \ldots, p_s$ only when it fulfils Condition (2).

Note. In the above procedure, the requirement of '80 percent' in Step (2) is based on our experimental results. As we will be using these polynomials in the construction of polynomials in Q, therefore, we want most of them to be such that the chosen ciphertext attack of Section 5.4 can be defeated.

Let us now use the instructions of this procedure to establish the following concrete example of a WGBC.

Example 4.3.6. Let $n = 3$ and consider the Weyl algebra

$$A_3 = \mathbb{F}_3[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$$

over the field of characteristic 3 and let the term ordering on B_n be $\sigma = \texttt{DegRevLex}$. We now introduce the following WGBC:

(1) **Secret Key:**
Choose the following polynomials of A_3

$$\begin{aligned} f_1 &= -\partial_1^3\partial_2^5\partial_3^5 - x_1^2x_2^3 + x_2^5 + 1, \\ f_2 &= x_2\partial_3^5 + \partial_1^3 - 1, \\ f_3 &= \partial_1^4\partial_2^5 + x_1^5\partial_2^7 - x_1^2\partial_2. \end{aligned}$$

Let $I = \langle f_1, f_2, f_3 \rangle$ be the ideal generated by these polynomials. Then the σ-Gröbner basis G of I is the set $\{g_1, \ldots, g_{11}\}$ where

$$\begin{aligned} g_1 &= \partial_1^3\partial_2^3, & g_2 &= x_2^4\partial_1^3 - \partial_3^5 + x_1^2x_2^2 - x_2^4, \\ g_3 &= x_1^2\partial_3^5 - x_1^4x_2^2 + x_2^6 + x_2, & g_4 &= x_1^2\partial_1^3, \\ g_5 &= \partial_2^3\partial_3^5 - x_1^2x_2^2\partial_2^3 + x_2^4\partial_2^3, & g_6 &= x_1^5\partial_2^6 - x_1^2, \\ g_7 &= x_2^9\partial_2^6 + x_1^4\partial_2^6 + x_1^2x_2^2\partial_2^6 + x_2^4\partial_2^6 - x_1x_2^5 - x_1, \end{aligned}$$

$$
\begin{aligned}
g_8 &= x_1x_2^7\partial_2^6 + x_1^3\partial_2^6 + x_1x_2^2\partial_2^6 - x_2^5 - 1,\\
g_9 &= \partial_3^{10} + x_2^3\partial_1^6 + x_2^3\partial_1^3 - x_1^2x_2 + x_2^3,\\
g_{10} &= x_1^2x_2^3 - x_2^5 - 1,\\
g_{11} &= x_2\partial_3^5 + \partial_1^3 - 1.
\end{aligned}
$$

The set G is our secret key and let $\mathscr{G} = (g_1, \ldots, g_{11})$. Note that, to fulfil Condition (2) in the above procedure we can use $g_2, g_3, g_5, g_7, g_8, g_9, g_{10}, g_{11} \in G$ for setting the message space and for creating the polynomials in the following public key.

(2) **Public Key:**
Let us now create public polynomial p_1, p_2, p_3 by choosing

$$
\begin{aligned}
h_{11} &= -x_1^5\partial_2^7 + x_2^3\partial_1^2\partial_3^5 - \partial_1^4\partial_2^5 + \partial_2\partial_3^5 - \partial_3^5 + x_1^2\partial_2 - 1,\\
h_{12} &= x_2^2\partial_1^5\partial_2^5\partial_3^5 + x_2\partial_1^5\partial_2^4\partial_3^5 - \partial_2^5\partial_3^5 + \partial_2^2 - \partial_2,\\
h_{13} &= -\partial_1^3\partial_2^5\partial_3^5 - \partial_1 + 1,\\
h_{21} &= x_2x_3^2\partial_2\partial_3^5 + x_1^2x_2^5\partial_2 + x_1^2x_2^3x_3^2\partial_2 - \partial_2\partial_3^5 - x_2^4 - x_2^3 + x_1^2\partial_2 - x_3^2\partial_2,\\
h_{22} &= x_1^4\partial_2\partial_3^5 + x_1^4\partial_2 - x_3^4\partial_2 + \partial_3^5 + 1,\\
h_{23} &= -x_1^2x_3^2\partial_2\partial_3^5 - x_1^6x_2^2\partial_2 - x_2^6x_3^2\partial_2 + x_1^2x_2^3 + \partial_2,\\
h_{31} &= x_2^7\partial_1^4\partial_2^7 - x_1^2x_2^4\partial_1\partial_2^6,\\
h_{32} &= -x_1x_2^7x_3^4\partial_1\partial_2^9 - x_2^5x_3^4\partial_1\partial_2^3 - x_1x_2^3\partial_2^3,\\
h_{33} &= -x_1^3x_3^4\partial_1\partial_2^6 - x_2^2\partial_1^7\partial_2 - x_1x_2^4\partial_3^5 + x_3^4\partial_1 + x_1x_2^3 - x_3^2\partial_1 + x_3\partial_1,\\
h_{34} &= x_3^4\partial_1\partial_2^6\partial_3^5 + x_2^4x_3^4\partial_1\partial_2^6 + x_2^6\partial_3^5 + x_1x_2\partial_1^4 + x_2\partial_3^5 - x_2^5 - x_1x_3^2\partial_2 +\\
&\quad x_1\partial_2 + x_3\partial_2 - 1,
\end{aligned}
$$

and then computing the standard forms of

$$
\begin{aligned}
p_1 &= h_{11}f_1 + h_{12}f_2 + h_{13}f_3,\\
p_2 &= h_{21}g_3 + h_{22}g_{10} + h_{23}g_{11},\\
p_3 &= h_{31}g_2 + h_{32}g_5 + h_{33}g_7 + h_{34}g_8.
\end{aligned}
$$

The polynomial p_1 has degree 20 and consists of 46 terms in its standard form. The polynomial p_2 has degree 14 and 51 terms and p_3 has degree 28 and 120 terms in its standard form. The polynomials h_{ij} are chosen such that the degree forms of the summands during the computation of the polynomials p_i cancel. In fact, the polynomials h_{ij} are chosen in the same way as described in (2) of Example 4.3.3. Moreover, the leading terms of the polynomials in G are not possible to guess from the polynomials p_1, p_2, and p_3 of the public key Q. These public polynomials are given in the Appendix C.2.

We set the public key $Q = \{\, p_1, p_2, p_3 \,\}$.

(3) **The Message Space:**

For the message space we choose

$$\mathscr{M} = \{x^{\alpha}\,\partial^{\beta} \;\mid\; |\alpha| + |\beta| \le 3\}$$

That is, $\langle \mathscr{M} \rangle_K$ is the vector space of all polynomials in A_3 of degree less than or equal to 3. With this $\mathscr{M}$, we can have 3^{84} possible plaintext messages. This message space is also known publicly.

(4) **Encryption:**

Suppose that the plaintext message $m \in \langle \mathscr{M} \rangle_K$ is given by the following polynomial

$$\begin{aligned} m \;=\; & x_1^2 x_2 - x_1^2 \partial_1 - \partial_1^2 \partial_2 + x_2 \partial_2^2 + \partial_3^3 + x_1 x_2 - x_2 x_3 - x_1 \partial_1 + x_3 \partial_1 + \\ & x_2 \partial_2 - \partial_1 \partial_2 - x_3 \partial_3 + \partial_2 \partial_3 - x_1 - x_2 + \partial_1 - \partial_3 + 1 \end{aligned}$$

For the encryption, choose

$$\begin{aligned} \ell_1 \;=\; & -x_1^6 x_2^9 x_3^6 \partial_1^5 \partial_2^8 \partial_3^4 - x_1^6 x_2^7 x_3^6 \partial_2^9 \partial_3^9 - x_2^7 x_3^9 \partial_2^7 \partial_3^7 + x_1 x_2^{10} x_3^4 \partial_2^{11} - x_2^6 x_3^5 \partial_1^2 \partial_2^7 \\ & + x_1 x_2^{10} \partial_1 \partial_2^6 + x_2^6 x_3^5 \partial_2^7 + x_3 \partial_1 - x_2 + \partial_1 \\ \ell_2 \;=\; & -x_1 x_2^4 x_3^4 \partial_1^7 \partial_2^{22} \partial_3^5 + x_1 x_2^6 x_3^2 \partial_1^8 \partial_2^{16} - x_2^8 x_3^2 \partial_1 \partial_2^{12} + \partial_1^2 \partial_2^5 \partial_3^5 + \partial_1^2 + \\ & \partial_1 \partial_2 - \partial_1 \partial_3 - x_2 - \partial_3, \\ \ell_3 \;=\; & x_1^6 x_2^4 x_3^2 \partial_1^{13} \partial_2 \partial_3^4 - x_1^7 x_2^4 \partial_1^7 \partial_2^{11} + x_1^3 x_2^8 \partial_1^7 \partial_2^{11} - x_1^7 x_2^2 x_3^2 \partial_1^7 \partial_2^{11} + \\ & x_1^3 x_2^6 x_3^2 \partial_1^7 \partial_2^{11} + x_1 x_2^5 \partial_1^5 \partial_2^4 + x_2^2 x_3^5 \partial_3^7 - x_1^2 x_2 \partial_1 \partial_3^5 + x_2 x_3 \partial_1^2 - \\ & x_2 x_3 - \partial_1 \partial_2 - x_1 \partial_3 - \partial_1 \partial_3 + \partial_1, \end{aligned}$$

and compute the ciphertext c as

$$c = \ell_1\, p_1 + \ell_2\, p_2 + \ell_3\, p_3 + m.$$

Then the polynomial c has degree 57 and there are 4289 terms in its standard form. We have selected the polynomials ℓ_1, ℓ_2, and ℓ_3 in such a way that the highest degree terms cancel and many other terms are either cancelled or their coefficients are changed in the middle and lower parts of the resulting ciphertext. For instance, choosing $-x_1^6x_2^9x_3^6\partial_1^5\partial_2^8\partial_3^4$ for ℓ_1 and then $x_1^6x_2^4x_3^2\partial_1^{13}\partial_2\partial_3^4$ for ℓ_3, cancels the term $-x_1^6x_2^{11}x_3^6\partial_1^{13}\partial_2^{13}\partial_3^9$ of degree 58 in c. Similarly, by choosing $-x_1x_2^4x_3^4\partial_1^7\partial_2^{22}\partial_3^5$ for ℓ_2, we get the leading term of the product $\ell_2\,p_2$ as $x_1^7x_2^{11}x_3^4\partial_1^7\partial_2^{23}\partial_3^5$ and then inserting $-x_1^7x_2^4\partial_1^7\partial_2^{11}$ in ℓ_3 cancels that leading term in c. To cancel the term $-x_1^6x_2^9x_3^6\partial_1^8\partial_2^{14}\partial_3^{14}$ of degree 57 so that it does not appear in c we insert the monomial $-x_1^6x_2^7x_3^6\partial_2^9\partial_3^9$ in ℓ_1. Continuing this way, we keep on adding and setting various terms for $\ell_1, \ell_2,$ and ℓ_3 and finally compute c as above. In this way, many terms in c are either cancelled or their coefficients are changed. The lower degree parts of the ciphertext polynomial c are dense enough to include many terms from the set $\mathscr{M}$. The monomials of the plaintext message m are either cancelled or their coefficients are changed in the ciphertext c. In fact, out of 18 monomials of m, 14 are not present in c.

(5) **Decryption:**
For recovering the plaintext message m we compute $\mathrm{NR}_{\sigma,\mathscr{G}}(c)$, the normal remainder of c modulo the Gröbner basis $\mathscr{G}$. An efficient implementation of the left Division Algorithm 2.3.18 recovers m within a second.

In the next chapter, we shall discuss the security of this instance of WGBC against known standard attacks.

Note. With reference to Procedure 4.3.5, note that why we are emphasizing that one has to be extra careful while attempting to create an instance of WGBC based on a randomly chosen left ideal of Weyl algebra. In the setting of Example 4.3.6, if we use the base field $K = \mathbb{F}_p$ $(p \geq 5)$ or $K = \mathbb{Q}$, then the Gröbner basis of the ideal I becomes $G = \{1\}$. For $\mathrm{char}(K) = 7$, the computation of Gröbner basis of the ideal $I = \langle f_1, f_2, f_3 \rangle$ takes 577.14 seconds on our computing machine and turns out to be $\{1\}$.

4.4 A WGBC Based on Remark 2.5.5

We shall now use the technique of Remark 2.5.5 for choosing an ideal in a Weyl algebra. We give an example for an instance of WGBC based on the following procedure:

Procedure 4.4.1. In the settings of Procedure 4.3.5, perform Step (1) as follows:

> Following the suggestions given in Remark 2.5.5, choose a left ideal I of A_n. Let the secret key $G = \{g_1, \dots, g_r\}$ be the reduced left σ-Gröbner basis of the ideal I.

Continue with Step (2) and (3) of Procedure 4.3.5 for choosing a message space $\mathscr{M}$ and constructing a public key Q.

We now illustrate this procedure by presenting the following instance of WGBC.

Example 4.4.2. Over the base field $K = \mathbb{F}_7$, we consider the Weyl algebra

$$A_3 = K[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$$

of index 3 and the term ordering $\sigma = \texttt{DegRevLex}$. Then we introduce the following WGBC.

(1) **Secret Key:**
Consider the Weyl polynomials given by

$$\begin{aligned} f_1 &= x_1^7, & f_2 &= x_1^3\partial_1^3 + x_1, \\ f_3 &= x_2^7, & f_4 &= x_2^2\partial_2^2 + x_2 + \partial_2, \\ f_5 &= x_3^7\partial_3^7 & f_6 &= \partial_3^4 + x_3, \end{aligned}$$

and let I be the left ideal $I = \langle f_1, f_2, f_3, f_4, f_5, f_6 \rangle$. Then the reduced left σ-Gröbner basis G of the ideal I consists of the following 11 polynomials:

$$\begin{aligned} g_1 &= x_1\partial_1^3 + 3x_1^2 + 3x_1\partial_1 - x_1 - 3\partial_1 - 3 \\ g_2 &= x_1^3 + 3x_1\partial_1^2 - 2x_1^2 - 2x_1\partial_1 - 2x_1 - 3\partial_1 + 2 \\ g_3 &= x_1^2\partial_1 + 3x_1^2 + x_1\partial_1 - 3x_1 - 2 \\ g_4 &= x_3^{10} - 2x_3^7\partial_3^2, \qquad g_5 = x_3^9\partial_3 - x_3^8, \end{aligned}$$

$$
\begin{aligned}
g_6 &= \partial_2^4+3\partial_2^3+2x_2^2-2x_2\partial_2+\partial_2^2-x_2-3\partial_2-3\\
g_7 &= x_3^7\partial_3^3-2x_3^9\\
g_8 &= x_2\partial_2^2-\partial_2^3-2x_2^2+2x_2\partial_2+2\partial_2^2+2x_2-3\\
g_9 &= x_2^2\partial_2+3\partial_2^3+2x_2^2-2x_2\partial_2+3\partial_2^2-2x_2-2\partial_2-3\\
g_{10} &= x_2^3-\partial_2^3+x_2^2-3x_2\partial_2+3\partial_2^2-2x_2-\partial_2-2\\
g_{11} &= \partial_3^4+x_3
\end{aligned}
$$

(2) **Public Key:**

For the public key Q, we compute the standard form of the polynomials

$$
\begin{aligned}
p_1 &= h_{11}g_1+h_{12}g_6+h_{13}g_4,\\
p_2 &= h_{21}g_2+h_{22}g_8+h_{23}g_5,\\
p_3 &= h_{31}g_3+h_{32}g_9+h_{33}g_7+h_{34}g_{11},
\end{aligned}
$$

by choosing

$$
\begin{aligned}
h_{11} &= x_1x_2^6\partial_2^3\partial_3^4-x_2^6\partial_2^3\partial_3^4-2x_2\partial_2^3\partial_3+3x_2\partial_2^3-2x_3^2\partial_3+x_2\partial_3-3\partial_3,\\
h_{12} &= -x_1^3x_2^6x_3^{10}\partial_1-3x_1^3x_2^3\partial_2^3\partial_3^4+2x_1x_2x_3\partial_1^2\partial_2^2\partial_3^4+\partial_1^3\partial_2^3\partial_3-x_1^3\partial_3^4+3x_3\partial_3^4\\
&\quad -x_1^3-3x_2^3+3x_3,\\
h_{13} &= x_1^3x_2^6\partial_1\partial_2^4+3x_1^3x_2^6\partial_1\partial_2^3+2x_1^3x_2^8\partial_1+x_1^3x_2^6\partial_1\partial_2^2+3x_1\partial_1\partial_2^2\partial_3^3+x_3\partial_3^3+\\
&\quad x_1\partial_1^2+x_2\partial_2^2-\partial_2^3+2x_1\partial_1+x_1\partial_2+3\partial_1,\\
h_{21} &= -3x_3^{11}\partial_1^2\partial_2^2\partial_3^2+3x_1^3x_3^2\partial_1\partial_3^2-3x_1^2\partial_3+2x_2\partial_1-x_3-\partial_2-2\partial_3,\\
h_{22} &= -3x_1^2x_2x_3^{11}\partial_2^3\partial_3^2-x_1^2x_2^2x_3^9\partial_2^3+x_1x_2^3x_3^2\partial_1^4\partial_3^3-2x_1\partial_1\partial_2^2\partial_3^2+x_2x_3\partial_1^2+\\
&\quad x_1\partial_1-2\partial_2^2-3x_3-\partial_3,\\
h_{23} &= 3x_1^2x_2^2x_3^2\partial_2^5\partial_3-3x_1^2x_2x_3^2\partial_2^6\partial_3+x_1^2x_2^3x_3^2\partial_2^3\partial_3-x_1^2x_2^2x_3^2\partial_2^4\partial_3-x_1^2x_2^4\partial_2^3+\\
&\quad x_1^2x_2^3\partial_2^4+2x_1^2x_2^2\partial_2^5+x_1^2x_2^2\partial_2^4+x_1^2x_2^3\partial_2^2+x_1^2x_2^2\partial_2^3+3x_1x_3\partial_3^2+3\partial_1\partial_2\partial_3\\
&\quad -2x_2+\partial_2,\\
h_{31} &= -3x_2x_3^{12}\partial_2\partial_3+3x_2x_3^{10}\partial_2^2\partial_3-3x_3^{10}\partial_2^3\partial_3-x_3^{10}\partial_2^2\partial_3+x_2^3x_3^9-\\
&\quad 2x_2x_3^7\partial_2\partial_3^3+3x_2x_3^9\partial_2-2x_2x_3^7\partial_3^3+x_1x_2x_3\partial_3+\partial_1^2\partial_2+3x_2\partial_3^2+\\
&\quad 2x_1\partial_3-3x_2-2\partial_3,\\
h_{32} &= -3x_1x_3^{10}\partial_1^3\partial_3+3x_1^2x_3^7\partial_3^3+3x_1x_3^7\partial_1\partial_3^3+2x_1x_3^3\partial_3^3+x_3^4-2x_1\partial_2^2-\\
&\quad \partial_2^3-3x_2\partial_3^2-x_2x_3+x_3^2+3\partial_1,
\end{aligned}
$$

$$\begin{aligned}
h_{33} &= 2x_1^2x_2x_3^3\partial_1\partial_2\partial_3 + x_1x_2^2x_3\partial_1^3 - x_1^2x_3\partial_2^2 + 3x_1x_3\partial_1\partial_2^2 + 2x_2x_3\partial_1\partial_2^2 - \\
&\quad 3x_1x_3\partial_2^3 - 2x_3\partial_1\partial_2^3 + x_1x_3\partial_3 - x_3\partial_2 - 3x_1\partial_3 - 2x_1, \\
h_{34} &= -2x_1^2x_2x_3^{10}\partial_1\partial_2 - x_1x_2\partial_1^3\partial_2\partial_3 - x_1x_2\partial_1^3\partial_3 - 2x_1^2x_2\partial_2^2\partial_3 - \\
&\quad 2x_1x_2\partial_1\partial_2^2\partial_3 + 2x_1^2\partial_2^3\partial_3 + 2x_1\partial_1\partial_2^3\partial_3 - 2x_1\partial_1^3\partial_3 - 3x_1^2\partial_2^2\partial_3 + \\
&\quad 3x_1x_2\partial_2^2\partial_3 + 3\partial_1^2 - 3x_2\partial_2 + x_1\partial_3 - 3x_3.
\end{aligned}$$

Then the Weyl polynomial p_1 has degree 23 and its standard form consists of 141 terms. The Weyl polynomial p_2 has degree 21 and there are 150 terms in its standard form and the polynomial p_3 has degree 18 and 204 terms. The public key is then the set[4] $Q = \{p_1, p_2, p_3\}$.

(3) **Message Space**
For the message space, we choose the K-vector space generated by

$$\mathscr{M} = \{x_1^{\alpha_1}x_2^{\alpha_2}x_3^{\alpha_3}\partial_1^{\beta_1}\partial_2^{\beta_2}\partial_3^{\beta_3} \mid |\alpha_1|, |\alpha_2|, |\beta_2| \le 1,\ |\alpha_3| \le 6,\ |\beta_1| \le 2,\ |\beta_3| \le 3\}.$$

There are 7^{672} different possible plaintext units and 10 polynomials in the secret key G have at least one term from $\mathscr{O}_\sigma(I) \setminus M$ in their supports.

(4) **Encryption:**
Let the plaintext message $m \in \langle \mathscr{M} \rangle_K$ be given by

$$\begin{aligned}
m &= x_2^2 - 2x_1\partial_1 - 3\partial_1^2 + 2x_1\partial_2 - 3x_2\partial_2 - 2\partial_1\partial_2 + \partial_2^2 - 2x_1\partial_3 - x_2\partial_3 + \\
&\quad x_3\partial_3 + 2\partial_1\partial_3 - 3\partial_2\partial_3 + 3\partial_3^2 + 2x_1 - 3x_2 - 2\partial_1 + \partial_2 + 3.
\end{aligned}$$

To encrypt this message m, we choose[5] sparse polynomials ℓ_1, ℓ_2, ℓ_3 of sufficiently high degree and compute the standard form of the Weyl polynomial

$$c = m + \ell_1 p_1 + \ell_2 p_2 + \ell_3 p_3.$$

For instance, let us encrypt m by choosing ℓ_1, ℓ_2 and ℓ_3 in the above repre-

[4] This set of polynomials is given in the Appendix C.2.

[5] These polynomials are chosen in the same way as described in the encryption part of Example 4.3.6.

sentation of c as follows

$$
\begin{aligned}
\ell_1 &= -3x_1^3x_2^3x_3^3\partial_1^8\partial_2^5\partial_3^5 - 2x_1^3x_2x_3^3\partial_1^8\partial_2^7\partial_3^5 + 2x_1^5x_3^5\partial_1^7\partial_2^8\partial_3 + 3\partial_2^2\partial_3 - 3\partial_2\partial_3,\\
\ell_2 &= -3x_1^6x_2^5x_3\partial_1^8\partial_2^7\partial_3 + x_1^6x_2^4x_3\partial_1^8\partial_2^5\partial_3^4 + 2x_1^6x_2^4\partial_1^8\partial_2^5\partial_3^4 + 3x_2^7x_3\partial_1^3\partial_2^4 +\\
&\quad x_3\partial_1^3\partial_3^2 - x_1x_2^2x_3\partial_1 - \partial_1^3\partial_3 - x_2^2,\\
\ell_3 &= 2x_1^5x_2^7\partial_1^6\partial_2^8\partial_3^6 + 3x_1x_2^9\partial_2^7 + 2x_2^5\partial_2^3\partial_3 + x_1x_2^2\partial_2^3\partial_3^2 + \partial_1.
\end{aligned}
$$

The resulting ciphertext c has degree 49 and its standard form consists of 6796 terms. Our choice of the polynomials ℓ_1, ℓ_2, and ℓ_3 has not only cancelled the degree form of $\ell_i p_i$ in c but also the lower part of ciphertext polynomial is dense enough to hide m completely. In fact, out of 18 monomials of m, 16 are not present in c.

(5) **Decryption:**
For recovering m we see that $m = \mathrm{NR}_{\sigma,G}(c)$. Therefore, for decryption, we have to compute normal remainder of the ciphertext polynomial c with respect to the Gröbner basis G. In this case, an efficient implementation of the Division Algorithm recovers m in a few seconds.

In the next chapter, we will study security issues of the concrete instances of WGBC presented in Examples 4.3.3, 4.3.6, and 4.4.2. We conclude this chapter by the following remark.

Remark 4.4.3. All instances of WGBC presented in this chapter are based on Weyl algebras over a finite field of positive characteristic. Although one can also attempt to construct an instance of WGBC based on the field $\mathbb{Q}$ of characteristic zero, based on the experimental results we recommend to use only the fields of characteristic $p > 0$. As we have seen in the observations in Example 4.1.4, the field $\mathbb{Q}$ is prone to coefficient-swell, and the growth of the support of polynomials can result in the requirement of large amount of memory for storing intermediate results during the computations. These phenomena may also result in unexpected size of the ciphertext polynomial and reduce the efficiency of the decryption process.

Chapter 5

Efficiency and Security

In this chapter we will consider the efficiency of the Weyl Gröbner Basis Cryptosystems. We also check the security issues of these systems against known standard attacks that are described in Chapter 3 and show that the instances of WGBC presented in Chapter 4 are secure against these attacks. We start by describing the efficiency of the computations that are involved when using the cryptosystem.

5.1 Efficiency

The efficiency of the WGBC strongly depends on the encryption and the decryption algorithms of the cryptosystem, that is, on the amount of work to be done by both *Alice* and *Bob* for secret communication over a public network. Therefore, in the setting of WGBC, both *Alice* and *Bob* have to be able to compute efficiently in the Weyl algebra A_n of index n over a field K of characteristic $p > 0$.

The two main operations involved in the encryption and the decryption processes of a WGBC are 'Weyl multiplication' and the computation of the 'normal remainder' modulo the secret key G. Efficient algorithms are available for performing these computations in Weyl algebras. These algorithms have been implemented in various computer algebra systems (see Section 2.6). We have also implemented these algorithms for the package `Weyl` of computer algebra system ApCoCoA. They can be used by calling the functions `Weyl.WMul()` and `Weyl.WNR()`, respectively. We refer to Appendix A for the description of these functions.

Another important operation involved in the process of secret communication is the transmission of the ciphertext message m over a public network. In the settings of a WGBC, the plaintext and the ciphertext units are the Weyl polynomials m and c respectively. We define the **data-rate** for transmitting a ciphertext unit over a network as the ratio of the size of the support of m to the size of the support of c. Moreover, the term **message expansion** refers to the length increase of a message when it is encrypted. One can measure the efficiency of Gröbner basis type cryptosystems either by the data-rate or by the message expansion. The message expansion can become a serious efficiency issue of such cryptosystem if the support of the resulting ciphertext grows too large as compared to the support of the plaintext unit m. It does not only affect the data-rate but also the storage and the decryption of the resulting ciphertext. In practice, it is very likely that, due to the way encryption is performed in such cryptosystems, the $\mathrm{Supp}(c)$ may become very large if various parameters are not properly restricted. For example, consider the Koblitz's "graph perfect code instance" of PCC presented in [25] (Ch. 5, §7), where the base ring is the commutative polynomial ring $P = \mathbb{F}_2[x_1, \ldots, x_n]$ in n indeterminates over the finite field $\mathbb{F}_2$. For security considerations, among other parameters, *Koblitz* suggested to use $n \approx 500$. In [23], the cryptanalysis of this instance of Polly Cracker is carried out. It is shown that even, by restricting n to 200, one gets a ciphertext polynomial containing more than $550,000$ terms in its support whereas, there is only one term in the support of m (see [23] for details). This, of course, results in a very bad data-rate for transmitting c. We shall now discuss the efficiency of WGBC in terms of the time required for the decryption and in terms of the data-rate for sending a ciphertext unit to its intended recipient.

In Chapter 2, we have noted that the multiplication of Weyl polynomials can increase the size of the resulting ciphertext given by the expression

$$c = \ell_1 p_1 + \cdots + \ell_s p_s + m.$$

This fact can reduce the efficiency of WGBC by decreasing the 'date-rate' for transmitting the ciphertext c over a network and also by decreasing the performance of the decryption process. The larger the size of the support of the ciphertext polynomial, the slower will the computation of *normal remainder* $\mathrm{NR}_{\sigma,\mathscr{G}}(c)$ with respect to the secret Gröbner basis $\mathscr{G}$ be. Of course, the efficiency of the decryption process also depends on the size and the number of polynomials in the secret key $\mathscr{G}$. On the basis of our experimental results, it has been observed that these issues are controllable in the setting of WGBC. This can be seen in the following table by observing the time (in seconds on our computing machine) taken by the decryption process and the data-rate for transmitting c, for our instances of WGBC presented in Examples 4.3.3, 4.3.6, and 4.4.2.

WGBC	Decryption	Data-Rate
Ex. 4.3.3	0.79	1/246
Ex. 4.3.6	0.59	1/238
Ex. 4.4.2	0.63	1/377

Table 5.1: WGBC: Decryption Time and Data-Rate

From the above table and many other similar instances of WGBC, we observe that instances of WGBC can be constructed which are efficient in terms of the time required by the decryption process. As far as the efficiency in terms of the data-rate is concerned, from the above table, we believe that the data-rates[1] achieved by instances of WGBC are manageable as compared to the instances of usual CGBC that have been presented so far. At the same time, as compared to usual CGBC, this nature of Weyl multiplication also gives WGBC additional security by hiding the coefficients of various terms of the plaintext in the above representation of c. Later, we will see that, since in the process of Weyl multiplication many new terms are introduced, it makes the "intelligent" linear algebra attack harder to apply on an instance of a WGBC.

Note that, from Proposition 2.1.5, the growth of the product of Weyl polynomials also depends on the characteristic of the underlying field K of A_n. That is, for fixed $f,g \in A_n$, the larger the characteristic of the base field K, the greater will the size of the support of the product $f\,g$ be. For characteristic $p = 0$, for example,

[1]These data-rates depend on the size of the support of the the message m. Depending on the size of the message space $\mathscr{M}$, the message m could have a larger support and this might result in a more better or similar data-rate as size of the Supp(c) may also increase for hiding various terms in Supp(m).

when $K = \mathbb{Q}$, the size of the support of this product will be a maximum. In particular, given two Weyl terms $t = x^{\alpha}\partial^{\beta}$ and $t' = x^{\alpha'}\partial^{\beta'}$, then from Proposition 2.1.5 it follows that the size of the support of the standard form of tt', together with the exponents β and α' also depends on the characteristic p of the underlying field K of the corresponding Weyl algebra. For fixed β and α' this size is maximal when $p = 0$. That is why, in the concrete instances of WGBC presented in Section 4.3, we have not used Weyl algebras over the field K of very large characteristic p.

In the next section we shall now discuss the security of WGBC against known standard attacks. In particular, we test our instances of WGBC presented in Section 4.3, by applying attacks based on linear algebra, the chosen ciphertext attack and the partial Gröbner basis attack.

5.2 Linear Algebra Attacks

In Section 3.3 we have described two attacks on PCC and CGBC, namely, the basic linear algebra attack and the "intelligent" linear algebra attack. In this section we briefly describe these attacks again in the setting of WGBC. We have implemented Attacks 3.5.1 and 3.6.1 in the setting of WGBC for the computer algebra system ApCoCoA[2]. We shall see that the instances of WGBC can be constructed that are secure against these attacks.

First we consider the basic linear algebra attack for WGBC. It is the same as Attack 3.5 for CGBC described in Chapter 3. For the sake of completeness, we rephrase it below in the setting of WGBC.

Attack 5.2.1. *Basic Linear Algebra Attack for WGBC*

Given an instance of WGBC, recall that the ciphertext polynomial c is constructed as follows:

$$c = m + \ell_1 p_1 + \cdots + \ell_r p_r.$$

In this representation of c, an eavesdropper, *Eve* knows the public polynomials $p_1, \ldots, p_r$ and the stolen ciphertext c. She also knows a set $\mathscr{M}$ containing the sup-

[2]see Appendix B.3 and B.4 for these implementations

port of m. Therefore, she can perform the following steps to attack the system using linear algebra.

(1) Fix an initial guess for the degree bound d_0 for the coefficient polynomials $\ell_1, \dots, \ell_s$ by setting $d_0 = d_c - d_p$.

(2) For $i = 1, \dots, s$,

  (i) Write down the polynomials ℓ_i as $\ell_i' = \sum_j a_{ij} t_j$ with indeterminate coefficients a_{ij}, where the sum ranges over all j such that the terms t_j are all terms of degree $\le d_0$.

  (ii) Write down the message m as $m_0 = \sum_j b_j t_j$ with indeterminate coefficients b_j, where the sum ranges over all j such that the terms t_j are the elements of $\mathscr{M}$.

(3) Compute the standard form of

$$c' = m_0 + \ell_1' p_1 + \cdots + \ell_r' p_r$$

to obtain a general ciphertext representation c' in the unknowns a_{ij} and b_j.

(4) Formulate a linear system of equations for the indeterminates a_{ij}, b_j by equating coefficients of c' to those of the original ciphertext c.

(5) Solve the above linear system of equations using linear algebra.

Case 1: If the system has a solution then recover the message m using the values b_j obtained from the solution of the system. That is, compute $m = m_0 = \sum_j b_j t_j$, and stop.

Case 2: If the system has no solution, then replace d_0 by $d_0 + 1$ and go to Step (2).

As in the case of CGBC, if the polynomials c and p_i are sparse, then the difficulty of the resulting problem of polynomial system solving increases as the number $d_c - d_p$ gets larger. In particular, one has to make the degree bounds $d_c - d_p$ large enough, in order to generate linear systems of equations in too many indeterminates to be solvable in an acceptable amount of time. At this point the first important difference between CGBC and WGBC stems from Proposition 2.1.5. As explained in the last section, the process of bringing $c = \ell_1 p_1 + \cdots + \ell_s p_s + m$ into standard form creates a large number of terms in the support of c. Hence the indeterminates a_{ij} appear in many different linear equations, and the linear equations are not sparse. Therefore, the user of a WGBC can make the resulting linear system of equation difficult to solve by selecting parameters n, d_c, d_p and d_ℓ appropriately.

By using an implementation of Attack 5.2.1, let us now examine how the instances of WGBC presented in Section 4.3 can be considered as secure against the basic Linear Algebra Attack.

Example 5.2.2. For the instance of the WGBC of Example 4.3.3, suppose that an attacker tries to recover the plaintext message m by using an implementation of the basic linear algebra attack. Note that in this case $d_c = 91$ and the public polynomials p_1 and p_2, have degrees 36 and 48 respectively. Therefore, the initial degree bound for the polynomials ℓ_1', and ℓ_2' is $d_0 = d_c - d_p = 55$. An implementation of Attack 5.2.1 on our 'computing machine' resulted in a dense linear system of size $3,183,545 \times 910,967$ which could not be solved. Moreover, because of the cancellation of the degree forms $\mathrm{DF}(\ell_i p_i)$, in c, for the success of the attack, an attacker has to solve even a larger linear system of equations.

Example 5.2.3. Let us now consider the instance of WGBC presented in Example 4.3.6. Before applying the attack, we determine the size of the linear system of equations that will be created by the basic linear algebra attack. In this case, we have $d_c = 57$, and the degrees of the public polynomials p_1, p_2, and p_3 are 20, 14, and 28 respectively. Therefore, to attack the system by using Attack 5.2.1, we have to start by assuming that the degrees of the polynomials ℓ_1', ℓ_2', and ℓ_3' are 37, 43, and 29 respectively. We also write the message m as a polynomial m_0 of degree less than or equal to 3 with indeterminate coefficients. With these informations, the basic linear algebra attack on this instance of WGBC will result in a linear system of equations of size $67,945,521 \times 21,703,514$. We believe that this system is infeasible to solve using the current known techniques of solving a dense as well as sparse linear system of equation over some finite field.

One can also similarly see that an attempt for breaking the cryptosystem presented in Example 4.4.2 by applying Attack 5.2.1 will be fruitless.

However, as described in Section 3.6, there is a more serious version of the basic linear algebra attack that is known as the "intelligent" Linear Algebra Attack [25]. The idea of the attack is to reduce the size of the linear system by reducing the number of unknowns in the linear system of equations obtained by the basic Linear Algebra Attack 5.2.1. Below we briefly describe this attack in our setting of WGBC and explain how WGBC can be made secure against it.

Attack 5.2.4. *Intelligent Linear Algebra Attack for WGBC*

Consider an instance of WGBC based on a Weyl algebra A_n. Let B_n be the set of all terms of A_n. Recall that, in the setting of WGBC, encryption is achieved by computing the standard form of

$$c = m + \ell_1 p_1 + \cdots + \ell_s p_s.$$

For $i = 1, \ldots, s$, write the coefficient polynomial ℓ_i as the polynomial ℓ_i' with indeterminate coefficients b_{ij}. Instead of using a dense representation of ℓ_i', compute the following set D.

$$D = \{t \in B_n \mid \exists t_p \in \bigcup_{i=1}^{s} \operatorname{Supp}(p_i), \text{ s.t. } t \cdot t_p = t_c \text{ for some } t_c \in \operatorname{Supp}(c)\}.$$

The set $D \subset B_n$ is the set of all the candidate terms for each ℓ_i.

Then use indeterminate coefficients b_{ij} in ℓ_i' only for the terms $t \in D$ and mount a linear algebra attack as described in Attack 5.2.1. That is, with these settings, one can tries to mount the attack on an instance of WGBC by following all the steps of Attack 3.6.1.

For the usual CGBC case, this attack might be very serious because of the fact that multiplication and addition of commutative polynomials rarely cancel terms completely. Moreover, as explained in Remark 3.6.2, this attack is more efficient when input polynomials are sparse. In the setting of WGBC, We have already explained in Section 5.1 that the process of converting $\ell_i p_i$ to standard form introduces many new terms in the ciphertext c and in turns reduces the sparsity of c. That is, the support of c becomes rather large and essentially all terms of suitable degrees pseudo-divide some term in $\mathrm{Supp}(c)$. Hence the set D in the Intelligent Linear Algebra Attack will contain a large number of candidate terms for the polynomial ℓ_i'. In other words, by a suitable choice of WGBC parameters given in Remark **??**, the user of WGBC can make it difficult to solve the linear system of equations obtained by using this attack.

Let us illustrate our claims with an extremely simple example.

Example 5.2.5. In the Weyl algebra $A_2 = \mathbb{F}_{31}[x_1, x_2, \partial_1, \partial_2]$, consider the polynomials

$$\begin{aligned} p_1 &= 2x_1^5\partial_1^2 + 4x_2^5 + 5x_1^3x_2 - 2x_1^2x_2^2 + 4x_1^3\partial_1 + 4x_2^2 + 3x_2\partial_1 - 2, \\ p_2 &= 33x_1^3x_2^3\partial_1^2\partial_2 + x_1^3x_2^4 + 4x_1^2\partial_1^2 + 8x_1^3 + 8x_1^2x_2 + 2x_2 + 3. \end{aligned}$$

Let us use the coefficient polynomials

$$\begin{aligned} \ell_1 &= -6x_2^4\partial_1^3\partial_2^5 + 10\partial_1^4 + 9\partial_1^3 + 8\partial_2^3 - \partial_2^2, \text{ and} \\ \ell_2 &= 4x_1^2x_2\partial_1^3\partial_2^4 - 6x_1\partial_1^3 - 12\partial_2^3 + 15\partial_1^2 + 14\partial_2^2 \end{aligned}$$

for the encryption. Notice that the numbers of terms in the supports of p_1, p_2, ℓ_1 and ℓ_2 are 8, 7, 5 and 5 respectively. The resulting ciphertext $c = m + \ell_1 p_1 + \ell_2 p_2$ has degree 11 and there are 184 terms in its standard form. However, in order to mount the intelligent linear algebra attack in this setting, the number of terms we have to consider for ℓ_1 and ℓ_2 is 268 each. This means that we have to solve a linear system of equations in more than 500 indeterminates. On the other hand, if the same set of polynomials are considered in the commutative polynomial ring $P = \mathbb{F}_{31}[x_1, x_2, \partial_1, \partial_2]$, then the intelligent linear algebra attack results in a linear system with 220 unknowns.

We have implemented Attack 5.2.4 for the computer algebra system ApCoCoA (see Appendix B.4) and tried to break the instances of WGBC presented in Section 4.3. We summarize our observations in the following examples.

Example 5.2.6. Consider the instance of WGBC given in Example 4.3.3 and apply the intelligent linear algebra attack using the ciphertext c, the public polynomials p_1, p_2 and the message space $\mathscr{M}$ as inputs. Note that, we have $d_c = 91$

and the public polynomials p_1 and p_2 have degrees 36 and 48 respectively. The total number of monomials in the public polynomials is 298. Therefore, the attack will start by initialising the degree $d = d_0 = 55$ for the polynomials ℓ'_1 and ℓ'_2 in unknown indeterminates b_{ij}. The next step is then to compute the set D for the candidate terms that are used for the encryption as explained above in the assumption of Attack 5.2.4. In this way, as compared to the basic linear algebra attack, the total number of unknowns b_{ij} reduces to 90,634 and we have to perform $(90,634 - 2808) \times 298 = 26,172,148$ Weyl multiplications of monomials for creating the general ciphertext c' in these unknowns. By comparing the coefficients of c' to those of c, the attack then results in a linear system of $368,344$ equations in $90,634$ unknowns. This task takes about 7 hours of CPU time on our computing machine. The next step is the setting-up of matrices for using linear algebra to solve this system. Another time consuming process of creating and filling up a large matrix of dimension $368,344 \times 90,634$ then starts. The resulting matrix contains 43,058,100 number of non-zero entries. We were unable to solve the system using the ApCoCoA package `LinBox` based on the C++ library of LinBox [16].

On the other hand, in these circumstances, if an attacker somehow is successful in solving this system by putting additional resources like using high-power computers and implementation of the attacks at lower level, he will learn that the system has no solution and that degree d_0 should be first increased to 56 and then to 57. Each time he has to try to solve even a larger system with more effort. With these observations, we believe that the instance of WGBC presented in Example 4.3.3 are to be hard to break by using intelligent linear algebra attack.

Remark 5.2.7. Because of the requirement (5) of Procedure 4.2.1, we note that, after bringing $c = \ell_1 p_1, \ldots, \ell_s p_s + m$ into standard form, the degree form $\mathrm{DF}(\ell_i p_i)$ cancel . An attacker does not know how many terms in the upper part of the ciphertext polynomial c are cancelled during this process. Therefore, the linear system of equation obtained by the first iteration of Attack 5.2.4 may not have any solution. That is depending on the number of terms cancelled in the upper-part of c, the attacker has to try solving more than one systems of linear equation, each time with more effort and resources. As we have seen in the above example that for recovering the plaintext message m, the attacker has to solve three very large systems of equations. Moreover, the users of WGBC can always make more difficult to solve the resulting linear system of equations. For instance, they can use the polynomials $\ell_1,\dots,\ell_s$ in such a way that makes the ciphertext dense in the lower and the middle parts.

For the instance of WGBC discussed in the above example, let us use the suggestions of choosing ℓ_i in the above remark and construct the following example.

Example 5.2.8. Consider again the instance of WGBC given in Example 4.3.3. Here we have the Weyl algebra $A_2 = \mathbb{F}_{13}[x_1,x_2,\partial_1,\partial_2]$ and the term ordering $\sigma =$ `DegRevLex`. The message m for sending using WGBC is given by

$$m = -6x_2^4\partial_2^3 + 6\partial_2^6 + 5x_2^4 - \partial_2^4 + 6x_2^3 + 6\partial_2^3 + x_1^2 + x_2\partial_2 - 3\partial_1\partial_2 + 2x_1 - 5$$

For encrypting m, we now choose different Weyl polynomials $\ell_1,\ell_2 \in A_2$ as follows:

$$\begin{aligned}
\ell_1 &= -5x_1^{10}x_2^{16}\partial_1^{12}\partial_2^{19} - 2x_1^8x_2^{18}\partial_1^{10}\partial_2^{21} - x_1^6\partial_1^{13} + \partial_1^{13} - 2\partial_2^{13} - 3x_1^5\partial_1^5 - x_1^5x_2^3 - \\
&\quad 3x_1^5 + x_1\partial_1 - 2x_2\partial_2 + \partial_1\partial_2 - \partial_1 + 1, \\
\ell_2 &= 4x_1^{11}x_2^{13}\partial_1^9\partial_2^{12} - 6x_1^9x_2^{15}\partial_1^7\partial_2^{14} - x_1^6\partial_1^{13} + \partial_1^{13} - 2\partial_2^{13} - x_1^5x_2^3 - \partial_1^5 + 4\partial_1^2\partial_2 + \\
&\quad x_1\partial_1 - 3x_2\partial_2 - 4\partial_1\partial_2 + x_2 + 2\partial_2 + 2.
\end{aligned}$$

With these ℓ_1 and ℓ_2, the new ciphertext polynomial $c = m + \ell_1 p_1 + \ell_2 p_2$ has degree 91 and there are 5278 terms in its standard form. The message m is also well-hidden, i.e. out of 12 monomials of m, 10 are not present in the ciphertext c. Again an efficient implementation of the normal remainder algorithm takes 2.7 seconds to decrypt the ciphertext. If an attacker tries to break the cryptosystem by using the intelligent linear algebra attack, then the attack starts with initial degree $d_0 = 55$ for the polynomials ℓ_1', ℓ_2' and results in a linear system 570,356 equations in 144,470 unknowns. This resulting system of equations is much harder to solve as compared to the linear system obtained by applying intelligent linear algebra attack on the ciphertext c of Example 4.3.3.

Remark 5.2.9. Although we were unable to solve the linear system resulting from the intelligent linear algebra attack on the instance of WGBC in the Example 4.3.3 and its modification in Example 5.2.8, we recommend to use a Weyl algebra of index $n > 2$ and choose the number of public polynomials $s > 2$ for achieving sufficient level of security against this attack. In fact, the larger the number of polynomials in public key, the larger will the number of unknowns in the resulting linear system be. This means that the linear system resulting from the intelligent linear algebra attack can always be made more difficult to solve by increasing the number of polynomials in the public key Q. This together with the suggestion given in Remark 5.2.7 provides us sufficient flexibility for making an attempt of mounting the intelligent linear algebra attack impractical.

Let us now observe how this attack behaves for the cryptosystems presented in Examples 4.3.6, and 4.4.2.

Example 5.2.10. Consider the instance of WGBC of Example 4.3.6. Note that, here we have the ciphertext polynomial c of degree 57 and its standard form consists of 4177 terms. In this setting, the attack starts with an initial degree of $d_0 = 43$ for the polynomials $\ell'_1, \ell'_2,$ and ℓ'_3 with unknowns b_{ij}. The set D of candidate terms for these polynomials contains 101,792 terms and the total number of monomials in all public polynomials is 217. Therefore, for the general ciphertext polynomial c' of degree 57, we have to perform $22,088,864$ Weyl multiplications of monomials. An implementation of this attack determines the size of the linear system required to solve is $5,872,648 \times 305,460$. Without setting up matrices for the corresponding system of equations, this task, took 47.3 hours of CPU time on our computing machine. We believe that this linear system of equations is very hard to solve by using current solving techniques. Therefore, we claim that this instance of WGBC is hard to break with the intelligent linear algebra attack.

Example 5.2.11. For the instance of WGBC presented in Example 4.4.2, we have Weyl algebra $A_3 = \mathbb{F}_3[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$ of index 3 and the term ordering $\sigma =$ `DegRevLex`. In this case, we have $d_c = 49$, the degree of the ciphertext c. The support of c contains 6798 terms. There are 495 total number of monomials in the polynomials $p_1, p_2,$ and p_3 and their minimal degree is 18. With these ingredients, the intelligent linear algebra attack fails to succeed for this instance of WGBC. In fact, in this case an attacker has to start with the initial guess $d_0 = 31$ for the polynomials ℓ'_i with unknowns b_{ij}. For the success of the attack, he has to solve a linear system of dimension $6,903,190 \times 640,083$, which, we believe, is extremely hard. The number of non-zero entries in the resulting matrix is 356,669,618.

Remark 5.2.12. Here we remark that the linear algebra type attacks, being single-break attack, have nothing to do with the secret key G. That is, if the attack is successful, the attacker would only be able to determine the plaintext message m corresponding to one stolen ciphertext c. The success of breaking one ciphertext does not reduce the amount of the time and the resources required to break another ciphertext.

All the above examples show that the instances of WGBC can be constructed to make them secure against the intelligent linear algebra attack. We believe that an attempt of trying to break an instance of WGBC by using these attacks is not practical. Note that the number of non-zero entries in the matrices of the linear systems of Examples 5.2.10 and 5.2.11 indicate that these matrices are sparse. Further investigation in this direction could be an attempt of exploiting the sparsity of these matrices for solving these linear systems in an efficient way. But is this practical? How difficult is it to accomplish? Are the corresponding matrices sparse enough that one can easily solve the system by exploiting the number of zero entries in these matrices? These are the questions that can only be answered by investigating 'structure' of these matrices and by studying all the techniques that have been developed so for solving 'sparse linear systems'.

We have not yet performed a detailed investigation for the possibility of such an attempt by using 'sparse linear algebra'. The techniques from the sparse linear algebra are efficient but most of the techniques depend on the structure of the corresponding matrices. In particular, the efficiency depends not only on the number of non-zero entries but also on their distribution in these matrices. Many techniques are designed only to work with the square matrices, i.e. with the determined systems and most of them are efficient for the symmetric matrices. We are interested in how efficient are the techniques for solving a sparse linear system when applied to the linear systems of Examples 5.2.10 and 5.2.11. On the other hand, if these systems are possible to solve by exploiting the sparsity of the system, we can always use suggestions of Remarks 5.2.7 and 5.2.9 such that mounting the intelligent linear algebra attack results in a linear system of even a more larger size. In this way we can make an attempt of using sparse linear algebra techniques more difficult to apply for the possibility of solving the resulting linear systems.

We illustrate it by the following example.

Example 5.2.13. Consider again the instance of WGBC of Example 4.4.2. In this case, the number of polynomials in the public key Q is $s = 3$ and the secret key G contains 11 polynomials $g_1, \ldots, g_{11}$. As suggested in Remark 5.2.9, we change the parameter s to 3 and construct two new polynomials p_4 and p_5 for the public key Q. In order to achieve this, let us choose

$$\begin{aligned}
h_{41} &= -3x_2^2x_3^4\partial_1^2\partial_2^4\partial_3^3 - 2x_3^4\partial_1^2\partial_2^6\partial_3^3 + 3x_1^3x_3^2\partial_1\partial_3^2 - x_1x_2\partial_1^3\partial_2\partial_3 - 2x_1^2x_2\partial_2^2\partial_3 - \\
&\quad 2x_1x_2\partial_1\partial_2^2\partial_3 + 2x_1^2\partial_2^3\partial_3 + 2x_1\partial_1\partial_2^3\partial_3 - 2x_1\partial_1^3\partial_3 - 3x_1^2\partial_2^2\partial_3 + 3x_1x_2\partial_2^2\partial_3 \\
&\quad -3x_1^2\partial_3 + 2x_2\partial_1 + 3\partial_1^2 - 3x_2\partial_2 + x_1\partial_3 + 3x_3 - 2\partial_3, \\
h_{42} &= 3x_1x_3^4\partial_1^5\partial_2^3\partial_3^3 + x_1x_2^2x_3\partial_1^3 - x_1^2x_3\partial_2^2 + 3x_1x_3\partial_1\partial_2^2 + 2x_2x_3\partial_1\partial_2^2 - 3x_1x_3\partial_2^3 \\
&\quad -2x_3\partial_1\partial_2^3 + x_1x_3\partial_3 - x_3\partial_2 - 3x_1\partial_3 - 2x_1, \\
h_{51} &= -3x_1^3x_2^3\partial_2^3\partial_3^4 - x_1^3x_2^3x_3^2\partial_2^3\partial_3 + \partial_1^3\partial_2^3\partial_3 - x_1^3\partial_3^4 + 3x_3\partial_3^4 - x_1^3 - 3x_2^3 + 3x_3, \\
h_{52} &= -3x_1^2x_2^2x_3^7\partial_2^3\partial_3^3 - x_1^2x_2^2x_3^9\partial_2^3 + x_1x_2^3x_3^2\partial_1^4\partial_3^3 - 2x_1\partial_1\partial_2^2\partial_3^2 + x_2x_3\partial_1^2 + x_1\partial_1 - \\
&\quad 2\partial_2^2 - 3x_3 - \partial_3, \\
h_{53} &= 3x_1^3x_2^3x_3^3\partial_2^3\partial_3 - x_1^2x_3\partial_2^2 + 3x_1x_3\partial_1\partial_2^2 + 2x_2x_3\partial_1\partial_2^2 - 3x_1x_3\partial_2^3 - 2x_3\partial_1\partial_2^3 + \\
&\quad x_1x_3\partial_3 - x_3\partial_2 - 3x_1\partial_3 - 2x_1.
\end{aligned}$$

and then compute the standard form of the polynomials

$$p_4 \;=\; h_{41}\,g_1 + h_{42}\,g_9, \text{ and } p_5 \;=\; h_{51}\,g_4 + h_{52}\,g_6 + h_{53}\,g_7.$$

The polynomial p_4 has degree 18 and contains 198 terms in its standard form. The degree of p_5 is 22 and there are 124 terms in its standard form. The public key is now $Q = \{p_1, p_2, p_3, p_4, p_5\}$. Let the message m be as given in Example 4.4.2. To encrypt the message m, together with ℓ_1, ℓ_2, ℓ_3 be as given in the above referred example, we also choose

$$\ell_4 \;=\; x_1^5x_2^8x_3^9\partial_1^4\partial_2^3\partial_3^2 + x_1^2x_2x_3^3\partial_1\partial_3^2, \text{ and } \ell_5 \;=\; -\partial_1^6 - 2\partial_2 - 2\partial_3,$$

and compute the ciphertext c as

$$c = m + \ell_1\,p_1 + \ell_2\,p_2 + \ell_3\,p_3 + \ell_4\,p_4 + \ell_5\,p_5.$$

With these changes, the resulting ciphertext c again has degree 49 and its support consists of 8410 terms. Moreover, the message m is well hidden. If we mount the intelligent linear algebra attack with the above 5 polynomials in the public key and the ciphertext c, then the resulting linear system has $1,544,445$ number of unknowns. Note here the difference in the number of unknowns with the corresponding number in Example 5.2.11. That is what we have explained in Remark 5.2.9 that by increasing the number of polynomials in the public key, one can always make it difficult to apply the intelligent linear algebra attack to the resulting instance of WGBC. Moreover, if we choose ℓ_4, ℓ_5 such that the degree d_c also becomes larger than 49, the degree of the ciphertext in Example 4.4.2, then the resulting linear system will become more difficult to solve. For instance, by choosing

$$\begin{aligned}
\ell_4 &= x_1^9 x_2^6 x_3^6 \partial_1^3 \partial_2^4 \partial_3^7 + x_1^5 x_2^8 x_3^9 \partial_1^4 \partial_2^3 \partial_3^2 - x_1^2 x_2 \partial_1^3, \text{ and} \\
\ell_5 &= -x_1^7 x_2^5 x_3^3 \partial_1^8 \partial_2^4 \partial_3^4 + x_1^7 x_2^4 x_3^3 \partial_1^8 \partial_2^5 \partial_3^4 + 2x_1^7 x_2^3 x_3^3 \partial_1^8 \partial_2^6 \partial_3^4 - 2\partial_2 - 2\partial_3,
\end{aligned}$$

the resulting ciphertext has degree $d_c = 52$ and 9267 terms in its support. In this setting, mounting the intelligent linear algebra attack, with the initial guess of $d_0 = 34$, results in a linear system in $2,247,150$ number of unknowns. Because of the cancellation of highest degree terms in c, an attacker will have to solve a very large linear system in more than 2.2 million indeterminate coefficients for the success of the intelligent linear algebra attack.

5.3 Partial Gröbner Basis Attack

We have described in Section 3.7 the partial Gröbner basis attack for the usual commutative Gröbner basis cryptosystem. The attack works exactly the same way for Weyl Gröbner basis cryptosystems as described in Attack 3.8. The obvious defence to this kind of attack is to choose the public polynomials $p_1, \ldots, p_s$ in such a way that the computation of **partial Gröbner bases** of the ideal $J = \langle p_1, \ldots, p_s \rangle$ is infeasible. In this section, we discuss the security of the instances of WGBC of Section 4.3 against a partial Gröbner basis attacks.

Recall that by a partial Gröbner basis H of the ideal $J = \langle p_1, \ldots, p_s \rangle \subset A_n$ upto the degree bound d we mean the output of the left Buchberger's Algorithm 2.3.24

modified such that each computation involving polynomials of degree higher than d is not performed. In the setting of WGBC an attacker can apply the *partial Gröbner basis attack* as follows:

Attack 5.3.1. *Partial Gröbner Basis Attack and WGBC*

Consider the Weyl algebra $A_n = K[x_1,\dots,x_n,\partial_1,\dots,\partial_n]$ of index n over K. Let σ be a degree compatible term ordering on B_n. Given an instance of WGBC based on A_n, let $J = \langle p_1,\dots,p_s\rangle$ be the ideal generated by the polynomials in the public key Q. For the partial Gröbner basis attack on WGBC, an attacker performs following steps.

(1) Choose a number $d > d_p$, where $d_p = \max\{\deg(p_i) | i = 1,\dots,s\}$.

(2) Compute a partial Gröbner basis H of J upto the degree bound d. Let $\mathscr{H}$ be the tuple of polynomials in H.

(3) Compute the normal remainder $m' = \mathrm{NR}_{\sigma,\mathscr{H}}(c)$. If m' is contained in the message space then stop otherwise replace d by $d+1$ and go to Step(2).

The probability of the success of the above Attack 5.3.1 increases with the increment in the degree bound d for H. In fact, it is more likely to succeed if $d = d_c$, the degree of ciphertext polynomial (see [8]). In [8], it is also suggested to start the attack by setting $d = d_c$ in the setting of CGBC. The question arises here: is this realistic? or is it always feasible to compute a partial Gröbner basis upto the degree bound $d = d_c$. In our setting of WGBC, the answer is NO. In fact for an instance of WGBC, there is a strong computational evidence that if the difference $d_p - d_c$ is greater than 25 then it is very likely that the computation of a partial Gröbner basis of J turned out to be infeasible. This claim is a consequence of Proposition 2.1.5. Even if we have an ideal I generated by randomly chosen sparse Weyl polynomials $f_1,\dots,f_k$ and plan to compute a partial Gröbner basis upto a degree bound d then at each step of the left Buchberger's Algorithm there is a considerable expansion in the supports of the resulting polynomials. This expansion of the supports not only increases the amount of the memory required to store the intermediate results but also affects the efficiency of computing the normal remainder of S-polynomials of very large sizes with respect to a set of polynomials with very large supports. In short, these facts slow down the entire computation enormously. We have already observed this behaviour of Buchberger's algorithm in Examples 4.1.2, 4.1.3, and 4.1.4.

In Procedure 4.2.1 for constructing a WGBC, we have explicitly requested that the designer checks that partial Gröbner bases of J are hard to compute for large degree bounds. As explained above, this is very easy to accomplish in the case of WGBC for a suitable choice of the parameter d_p and the polynomials h_{ij} used for creating the public polynomials $p_1,\ldots,p_s$. Of course, our polynomials $p_1,\ldots,p_r$ are not entirely random, since they are contained in a larger ideal which has a simple Gröbner basis, namely G. But we have not been able to use this fact to the benefit of the attacker, and in all cases that we tried, the predicted expansion of the supports happened indeed. The success of Attack 5.3.1 highly depends on the successful computation of a partial Gröbner basis of the ideal $J = \langle p_1,\ldots,p_s\rangle$ for large degree bounds. From all our experimental results we believe that in the setting of WGBC, if the difference $d_c - d_p$ is kept greater than 25 then the success of the partial Gröbner basis attack cannot be guaranteed because of the above explanations. In the following examples we examine the security of the instances of WGBC presented in Section 4.3 against the partial Gröbner basis attack.

Example 5.3.2. Consider the WGBC presented in Example 4.3.3 and let $J = \langle p_1, p_2\rangle$ be the ideal generated by the polynomials in the public key. Note that we have $d_c = 91$ and $d_p = \max\{36,48\} = 48$ therefore, to start the attack we set the degree bound $d = 60$ for computing a partial Gröbner basis of J. Using the CAS `Singular` on our computing machine, we computed a partial Gröbner basis H of the ideal J in 3613.93 seconds of CPU time. The set H contains 108 polynomials consuming 183 MB of memory. The reduction of the ciphertext c modulo H returns a remainder with $284,745$ terms. This process takes 17547.56 seconds of CPU time on our computing machine. As required by Attack 5.3.1, we replaced d with $d+1 = 61$ and continue. For $d = 65$, we were unable to compute a partial Gröbner basis of the ideal J in 546513.6 seconds (151.81 Hours) of CPU time. At this point, the computation was progressing very slow and the amount of memory consumed during the computation was 3481.6 MB. For the possible success of the attack, one has to compute a partial Gröbner basis of J for the degree bound $d \geq 91$. With these observations, we claim that the computation of a partial Gröbner basis for the success of the partial Gröbner basis attack is infeasible.

Example 5.3.3. Consider now the instance of WGBC presented in Example 4.3.6. In this case we have $d_c = 57$ and $d_p = 28$. For attacking the system with Attack 5.3.1, let us choose $d = 45$. Let $J = \langle p_1, p_2, p_3 \rangle$ be the ideal generated by the polynomials in the public key Q of the cryptosystem under consideration. With these ingredients, the computation of a partial Gröbner basis H of J for $d = 45$ takes $136,401.80$ seconds on our computing machine. The resulting set H contains 195 Weyl polynomials the amount of memory required to store these polynomials grows to 12.1 GB. Note here the expansion in the supports of the resulting polynomials. We interrupted the process of computing the normal remainder of c with respect to H after 18,921 minutes of CPU time to stop without any output. During this process the the intermediate results had grown enough to consume more than 16 GB of the system memory. We then started to compute a partial Gröbner basis with the degree bound $d = 47$ and could not compute H. In fact, we interrupted the computation after more than 7 days of CPU time on our computing machine to terminate without an output. At the time of interruption, the computations had already consumed 16.3 GB of memory and was progressing very slow. Hence there is a significant computational evidence that the partial Gröbner basis attack fails for this instance of WGBC.

In the following we illustrate how the partial Gröbner basis attack fails when applies to the instance of WGBC of Example 4.4.2.

Example 5.3.4. Consider the case of WGBC presented in 4.4.2. The given Weyl algebra is $A_3 = \mathbb{F}_7[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$ and the term ordering is `DegRevLex`. Moreover, we have $d_c = 49$ and $d_p = 23$. Let $J = \langle p_1, p_2, p_3 \rangle$ be the ideal generated by the polynomials in the public key Q. In this case we fail to compute a partial Gröbner basis of J due to very fast growth of memory required for the computations. For instance, using the CAS `Singular`, we set the degree bound $d = 32$ for computing a partial Gröbner basis H of the ideal J. The computation of H takes 38884.19 seconds on our computing machine. The set H contains 326 polynomials and fails to reduce the ciphertext c in 15065 minutes of CPU time. We were unable to compute a partial Gröbner basis of J for a degree bound $d > 32$ using our current resources. With these observations, we claim that the instance of this WGBC is secure against the partial Gröbner basis attack.

Notice that in all above examples the attempts of trying to break the instances of WGBC by partial Gröbner basis attack fail. In fact, in all these cases the computation of a partial Gröbner basis for a degree bound $d = d_c$ is infeasible. Moreover, if H is a successfully computed partial Gröbner basis of the ideal $J = \langle p_1, \dots, p_s \rangle$ for some degree bound d such that $d_p < d < d_c$, then the normal remainder of the ciphertext c with respect to H is not contained in the message space $\mathscr{M}$.

5.4 Chosen Ciphertext Attack and WGBC

Recall the chosen ciphertext attack explained in the Section 3.9 for the usual CGBC. In the setting of WGBC, one can apply the chosen ciphertext attack exactly the same way as described for the CGBC setting in Attack 3.9.1. That is, the attacker *Eve*, should have a temporary access to the decryption black box for decrypting a finite number of ciphertext messages of her choice. For $i = 1, \dots, r$, let us write 'secret' polynomials g_i in the secret key G as:

$$g_i \;=\; t_i + h_i, \;\text{ with }\; \mathrm{LT}_\sigma(h_i) <_\sigma t_i$$

In order to attack an instance of WGBC, *Eve* should also know or be able to guess the leading terms t_i of the polynomials $g_i \in G$. With this knowledge, she can then construct a 'fake' ciphertext message of the form

$$c_i' = t_i + \sum_j h_{ij}' p_j.$$

By using her temporary access to the decryption black box, she decrypt the fake ciphertext message c_i'. As a result, for each $i = 1, \dots, r$, she will get $\mathrm{NR}_{\sigma,\mathscr{G}}(c_i) = -h_i$. Then by recombining she will find all secret polynomials $g_i = t_i + h_i$. This reveals the complete secret key G of the corresponding cryptosystem. This attack works well both on the basic set-up of CGBC and Rai's basic non-commutative Polly

Cracker cryptosystem because their decryption processes are not able to distinguish such fake ciphertext messages from the original one. To defend this attack in the setting of non-commutative Polly cracker cryptosystem, Rai and Bulygin [42] have proposed the following countermeasures:

(1) Do not publish the complete set $\mathscr{O}_\sigma(I)$. Publish only a (small) part $\mathscr{M} \subset \mathscr{O}_\sigma(I)$ and use $\mathscr{M}$ as a K-basis for the message space.

(2) Ensure that the *tail* h_i of each polynomial $g_i \in G$ contains at least one term from $\mathscr{O}_\sigma(I) \setminus \mathscr{M}$ in its support. In this way, if the attacker guesses $\mathrm{LT}_\sigma(g_i)$ and tries to decrypt it, countermeasure (3) will make sure that he fails.

(3) Design the decryption algorithm such that if the normal remainder of the ciphertext c is not contained in $\langle \mathscr{M} \rangle_K$ then either return an error message or the original ciphertext without reduction. In this way, when the attacker decrypts a term outside $\mathscr{M}$, the term is returned unchanged and no secret information is revealed.

These countermeasures are suggesting us a way of recognising illegal or fake ciphertext messages and hence the above explained chosen ciphertext attack will not work. That is, if the decryption algorithm computes a normal remainder which is not contained in $\langle \mathscr{M} \rangle_K$, it is clear that an illegal ciphertext was used. Therefore the decryption algorithm does not reveal the normal remainder, but returns the ciphertext unchanged. It has been argued that countermeasure (1) reduces the efficiency of the cryptosystem too much. By restricting $\mathscr{M}$ to a proper subset of $\mathscr{O}_\sigma(I)$ we can make the probability for a random polynomial to be a valid ciphertext as small as we like.

The above explained countermeasures can be adapted for any Gröbner basis type cryptosystem. Since WGBC is a special case of GBC, we have already proposed to design a WGBC in a way that its basic set-up automatically recognises the illegal ciphertext messages. For instance, notice that in the introduction of the WGBC in Cryptosystem 4.1.1, we have adapted countermeasures (3). The other two countermeasures are part of the set-up proposed in Procedure 4.2.1.

Note also that all the instances of WGBC presented in Section 4.3 have resistance against chosen ciphertext attack from the procedures on which they are based.

5.5 Adaptive Chosen-Ciphertext Attack

In [25] *Koblitz* described an *adaptive chosen ciphertext attack* for PCC (see Chapter 5, §3, Exercise 11) which exploits the fact that PCC is homomorphic. That is, if $c, c' \in P$ are ciphertext units corresponding to the plaintext messages m and $m' \in K$ respectively, then it holds that $c + c'$ and $c \cdot c'$ are ciphertext units for $m + m'$ and $m \cdot m'$ respectively. *Koblitz* described this attack as follows:

Suppose that two companies A (*Alice*'s company), and C (*Cathy*'s company) are communicating with B (*Bob*'s company) using Bob's public key. On many questions, C is cooperating with B, but there is one extremely important customer who is taking competing bids from a group of companies led by A and B, and from a different consortium led by C. C knows that A has just sent B the encrypted amount of their bid, and she desperately wants to know what it is. Suppose that A's message m is sent as the ciphertext c, and that *Cathy* is able to see it. *Cathy* creates a ciphertext, $c' = c_0 + c + m_0$ where $c_0 = \sum_{i=1}^{s} h_i p_i$ is an encrypting polynomial, and c' decrypts to the element m' of the message space $\mathscr{M}$. She sends c' to B, supposedly part of a message on an unrelated subject. She then informs B that she had a computer problem, lost her plaintext, and thinks that an incomplete sequence of bits was encrypted for Bob. Could *Bob* please send her the decrypted m' that she obtained from c', so that *Cathy* can reconstruct the correct message and re-encrypt it? Since c_0 vanishes during the decryption process, and c decrypts to m, it follows that c' decrypts to $m' = m + m_0$. Hence m' can be used to find $m = m' - m_0$. *Bob* is willing to give *Cathy* m' because he is unable to see any connection between c' and c or between m' and m, and because *Cathy*'s request seems reasonable when they are exchanging messages about a matter on which they are cooperating.

Note that the way c' is constructed makes it a legitimate ciphertext and there seems to be no straightforward way for *Bob*'s decryption algorithm to recognize it as a security threat. Even with the countermeasures presented in Section 5.4 for the chosen-ciphertext security, one cannot recognize such a fake ciphertext message. Moreover, the attack in this form is a single-break attack since the message corresponding to only one ciphertext can be recovered at a time and it has nothing to with the secret key.

In the following we summarise this attack in our setting of WGBC and then provide countermeasures for the security of the instances of WGBC against this attack.

Attack 5.5.1. *Adaptive Chosen-Ciphertext Attack*

Let *Alice* and *Cathy* be communicating with *Bob* using a WGBC. Suppose that *Cathy* knows the ciphertext $c = m + \sum_{i=1}^{s} \ell_i p_i \in A_n$ that *Alice* has just sent to *Bob*. As explained above, *Cathy* has decided to cheat *Bob* to break the ciphertext c. In order to recover the plaintext m corresponding to c she has to perform the following steps.

(1) Create a fake ciphertext message c' as $c' = c_0 + c + m_0$, where $c_0 = \sum_{i=1}^{s} \ell_i p_i \in \langle p_1, \ldots, p_s \rangle$ and $m_0 \in \mathscr{M}$.

(2) Request *Bob* to decrypt c' and send the result m' to her. Note that

$$m' = \mathrm{NF}_{\sigma,\mathscr{G}(c')} = \mathrm{NF}_{\sigma,\mathscr{G}}(c_0) + \mathrm{NF}_{\sigma,\mathscr{G}}(c) + \mathrm{NF}_{\sigma,\mathscr{G}}(m_0) = m + m_0.$$

(3) Recover the plaintext message m as $m = m' - m_0$.

In [42], *Rai* and *Bulygin* have proposed a countermeasure to overcome the above attack in the setting of *Rai*'s non-commutative Polly Cracker cryptosystem. Because of the richness of the WGBC message space $\mathscr{M}$, the countermeasure of [42] (see Countermeasure 4.3) can also be adapted for the security of WGBC against Attack 5.5.1. This countermeasure works as follows:

(1) *Bob*'s public key is $Q = \{p_1, \ldots, p_s\}$ and he sets his secret key G such that the message space $\mathscr{M}$ should be large enough to be partitioned into disjoint subsets.

(2) *Bob* chooses *Alice*'s message space as $\mathscr{M}_A \subset \mathscr{M}$ and *Cathy*'s message space as $\mathscr{M}_C \subset \mathscr{M}$ such that $\mathscr{M}_A \cap \mathscr{M}_C = \emptyset$.

(3) Design the decryption algorithm to recognize the ciphertext by its sender.

In this way, *Bob* can easily recognize *Cathy*'s fake ciphertext of the form $c' = c_0 + c + m_0$, where c is the ciphertext used by *Alice* to encrypt the message $m \in \mathscr{M}_A$. Let $m' \in \mathscr{M}$ be the decryption of c'. Since both $\mathscr{M}_A$ and $\mathscr{M}_C$ are publicly known, if $m_0 \in \mathscr{M}_C$ then m' does not belong to $\mathscr{M}_A$ as well as $\mathscr{M}_C$ and decryption algorithm will return an error message about the suspicious nature of *Cathy*'s ciphertext. On the other hand, if $m_0 \in \mathscr{M}_A$, then m' will be an element of $\mathscr{M}_A$ and again decryption algorithm will recognize that an invalid ciphertext is sent by *Cathy*. Hence by adapting the countermeasures presented in [42], one can overcome Attack 5.5.1. An other technique to defend the attack is described in L. Van Ly thesis [35](see §4,4). A similar countermeasure can also be adapted in the setting of WGBC. We, therefore, believe that this attack does not appear to be a major threat for the security of WGBC. Further study of these cryptosystems might also results in other more interesting and efficient techniques for the chosen-ciphertext security of WGBC.

5.6 Further Security Parameters

In this section we will describe how additional security of WGBC can be achieved. In [51] it has been pointed out that for sending a message m to *Bob* by using a CGBC, *Alice* has nothing to do with the characteristic p of the underlying field K and the term ordering σ. Therefore, one can achieve additional security by hiding the characteristic p of the field K and the term ordering σ on the terms of the base ring from the public information of CGBC. For the case of the usual Polly Cracker cryptosystems, this suggestion has been worked out in detail in [51]. This suggestion can also be adapted for the case of WGBC for making the cryptosystem even more secure.

Remark 5.6.1 (Make p and σ secret)**.** Here we remark that one can achieve additional security by hiding the characteristic p of the field K and the term ordering σ on A_n from the public information of **WGBC**. By keeping p and σ secret,

- we increase the cost of linear algebra attack.
- the chosen cipher text attack will not be possible in general settings.
- for the Gröbner basis computation of the public ideal J, the attacker has to guess for a true p and the term ordering σ on A_n.

Chapter 6

Two Sided Weyl Gröbner Basis Cryptosystems

In Chapter 4, we have presented several concrete instances of our proposed left Weyl Gröbner basis cryptosystems and in Chapter 5, we have discussed the security of these instances of WGBC against known standard attacks. We have strong computational evidence that these concrete instances of WGBC have resistance against these attacks. On the other hand, we are also aware of the possibility of modifying the attacks that are based on linear algebra. Such improvements might be possible by introducing some more clever strategies or by playing with the statistics of the terms in the ciphertext and the public key polynomials for reducing the size of the resulting linear system of equations to solve it in a reasonable time. Success of these attacks is also based on the current available techniques for solving a system of linear equations. Although we were unable to break our instances of WGBC by using the intelligent linear algebra attack, we are still interested in 'totally' avoiding the attacks based on linear algebra. This objective can be achieved by choosing proper two-sided ideals in Weyl algebras and then construct a GBC based on these ideals. We shall call such a system a *Two-sided Weyl Gröbner Basis Cryptosystem* (TWGBC).

In this chapter, we describe two-sided ideals of Weyl algebras and explain how we can compute a two-sided Gröbner basis of such ideals. We shall then introduce TWGBC in Section 6.2. These cryptosystems are based on the difficulty of computing two-sided Gröbner bases in Weyl algebras over fields of positive characteristic. We shall also present some concrete instances of such cryptosystems and discuss their security and efficiency issues.

6.1 Two-Sided Gröbner Bases

Let us first recall some definitions from non-commutative polynomial ring theory.

Definition 6.1.1. Given a non-commutative ring R, we say that a subset $I_T \subset R$ is a **two-sided ideal** of R if I_T is closed with respect to addition and for any $\ell, r \in R$ and $f \in I_T$ we have $\ell\, f\, r \in I$.

Definition 6.1.2. Given a subset $F \subset R$ of a a ring R, we say that $\langle F \rangle_T$ is the **two-sided ideal generated by** F if it is of the form

$$\langle F \rangle_T = \{ \sum_{i \in \Lambda} \ell_i f_i r_i \mid \ell_i,\, r_i \in R,\, f_i \in F, \Lambda \text{ finite} \}$$

Moreover, a two-sided ideal I_T is called trivial if $I_T = \{0\}$ or $I_T = R$ and otherwise it is called non-trivial.

We shall now describe some two-sided ideals of the Weyl algebra A_n of index n. Recall that the Weyl algebra $A_n = K[x_1, \ldots, x_n, \partial_1, \ldots, \partial_n]$ of index n over the field K is *simple* when K has characteristic 0. That is, A_n does not have any non-trivial 2-sided ideals if $\mathrm{char}(K) = 0$. On the other hand, if $\mathrm{char}(K) = p > 0$, then this property does not hold anymore. This follows immediately from the following example.

Example 6.1.3. Consider the Weyl algebra $A_1 = \mathbb{F}_p[x, \partial]$ of index 1 over the finite field $\mathbb{F}_p$ of prime characteristic p. Take the element $\partial^p \in A_1$. For any term $t = x^\alpha \partial^\beta \in A_1$, we have, from Proposition 2.1.5 that

$$\begin{aligned}
\partial^p t &= (\partial^p x^\alpha)\partial^\beta \\
&= \left(\sum_{j=0}^{\min\{p \mod p,\, \alpha \mod p\}} j! \binom{p}{j} \binom{\alpha}{j} x^{\alpha-j} \partial^{p-j} \right) \partial^\beta \\
&= \left(\sum_{j=0}^{0} j! \binom{p}{j} \binom{\alpha}{j} x^{\alpha-j} \partial^{p-j} \right) \partial^\beta \\
&= (x^\alpha \partial^p)\partial^\beta \;=\; x^\alpha(\partial^p \partial^\beta) \;=\; (x^\alpha \partial^\beta)\partial^p \;=\; t\,\partial^p
\end{aligned}$$

It follows that ∂^p commutes with every term $t \in A$. Therefore, $I = \langle \partial^p \rangle$, the left ideal generated by ∂^p, is also a two-sided ideal of A_1. Hence A_1 is not simple.

In fact, for the Weyl algebra A_n over a field $K = \mathbb{F}_p$ of positive characteristic p we have Proposition 2.2.9. It states that, if A_n is a Weyl algebra of index n over a field K of positive characteristic $p > 0$, then the center C_n of A_n is a commutative polynomial ring in $2n$ indeterminates over K and it is given by

$$C_n = K[x_1^p, \ldots, x_n^p, \partial_1^p, \ldots, \partial_n^p].$$

In view of this proposition and the above example, we note that, for the Weyl algebra $A_n = \mathbb{F}_p[x_1, \ldots, x_n, \partial_1, \ldots, \partial_n]$, if I is the left ideal generated by the elements in the set $\{x_1^p, \ldots, x_n^p, \partial_1^p, \ldots, \partial_n^p\}$, then it is also be a two-sided ideal. In particular, any non-trivial left ideal I of A_n whose system of generators is contained in the center C_n is always a two-sided ideal of A_n.

From now on, we let $K = \mathbb{F}_p$ be a field of positive characteristic p and let A_n be the Weyl algebra of index n over the field K. By an ideal we mean a two-sided ideal of the Weyl algebra A_n and we denote it by the symbol I_T unless otherwise specified. The K-vector space basis of A_n as defined in Section 2.1 is the set B_n of all terms given by,

$$B_n = \{x^\alpha \partial^\beta \mid \alpha, \beta \in \mathbb{N}^n, n \geq 1\}. \tag{6.1}$$

Example 6.1.4. Consider the following Weyl algebra

$$A_2 = \mathbb{F}_{13}[x_1, x_2, \partial_1, \partial_2]$$

of index 2 over the finite field of characteristic 13. Then the center C_2 of A_2 is given by

$$C_2 = \mathbb{F}_{13}[x_1^{13}, x_2^{13}, \partial_1^{13}, \partial_2^{13}].$$

The following are some non-trivial two-sided ideals of A_2:

$$\begin{aligned}
I_{T_1} &= \langle x_1^{13}, x_2^{13}, \partial_1^{13}, \partial_2^{13} \rangle_T \\
I_{T_2} &= \langle x_1^{13} - 1, \partial_1^{13} - 3, 2\partial_2^{13} - 5 \rangle_T, \\
I_{T_3} &= \langle x_1^{13} x_2^{13} - 1, \partial_1^{13} \partial_2^{13} - 5 \rangle_T \\
I_{T_4} &= \langle x_2^{26} - \partial_1^{13} \partial_2^{13} - 3, x_1^{13} x_2^{13} \partial_1^{13} - 3\partial_2^{13} - 1 \rangle_T \\
I_{T_5} &= \langle \partial_2^{13} \rangle_T, \text{ a principal two-sided ideal}
\end{aligned}$$

Example 6.1.5. For the Weyl algebra $A_4 = \mathbb{F}_2[x_1,x_2,x_3,x_4,\partial_1,\partial_2,\partial_3,\partial_4]$ of index 4 over the field $\mathbb{F}_2$ of characteristic 2, the center C_2 is given as

$$C_4 = \mathbb{F}_2[x_1^2,x_2^2,x_3^2,x_4^2,\partial_1^2,\partial_2^2,\partial_3^2,\partial_4^2].$$

The following are non-trivial two-sided ideals of A_2

$$\begin{aligned} I_T &= \langle x_1^4x_2^4\partial_3^4\partial_4^2 - \partial_1^4\partial_3^2 - x_4^2 - 1,\ \partial_1^4\partial_4^2 - x_2^2x_3^2 + x_4^2 + \partial_1^2 - \partial_4 + 1\rangle_T \\ J_T &= \langle x_1^6\partial_1^4 - x_2^4\partial_2^6 + x_3^4\partial_3^2 + x_4^2\partial_4^2 + 1,\ \partial_1^6\partial_2^4 - \partial_3^6\partial_4^2 + 1, \\ &\quad x_1^4x_4^4 - x_2^4x_4^2 - x_3^2 - \partial_1^2 + \partial_3^2 - 1\rangle_T \end{aligned}$$

Note that each term in the support of the generating polynomial of the above ideals belongs to the center C_4.

Remark 6.1.6. For a two-sided ideal $I_T \subset A_n$, if its generating system is contained in the center C_n then it does not mean that all the elements of I_T commute. For instance, in the Weyl algebra $A_1 = \mathbb{F}_3[x,\partial]$, the ideal $\langle x^3\rangle_T$ is a two-sided principal ideal generated by $x^3 \in A_1$. Here $x^3 \in C_n$ and the element $\partial(x^3)x = x^3(\partial x) = x^3(x\partial + 1) = x^4\partial + x^3$ belongs to I_T but it is not contained in C_n.

We shall now briefly explain the theory of two-sided Gröbner bases of two-sided ideals of the Weyl algebra A_n by following the approach of [24] or [26] and compute two-sided Gröbner bases using the algorithm presented in [30].

Given a non-empty subset $F \subset A_n$, we denote the left, right and two-sided ideals generated by F by $\langle F\rangle_L$, $\langle F\rangle_R$, and $\langle F\rangle_T$ respectively. Recall from Section 2.3, we consider a left-sided generating system as the set of left-sided generators of a left-sided ideal and compute its left Gröbner basis by using left Division Algorithm 2.3.18 and left Buchberger Algorithm 2.3.24. In the same way one can also compute a right Gröbner basis of a right ideal by using the right multiplication instead of the left in these algorithms. The approach used in [24] and [26] for two-sided Gröbner bases is that, unlike the one-sided case, we consider consider a given two-sided generating system as a left or right sided generating system equivalent to the given two-sided one. That is, given a two-sided ideal I_T, and a term ordering σ, then I_T, being a two-sided ideal is also a left ideal of A_n. Therefore, from Chapter 2, Section 2.3 it has left σ-Gröbner basis G_L. We can compute G_L by using the Buchberger Algorithm 2.3.24. Then, for computing a two-sided Gröbner basis of I_T, we can for example start from the left Gröbner basis G_L, and complete it successively to the right structure, keeping the left one (see [30], Ch. 2 §3).

Definition 6.1.7. Let σ be a term ordering on A_n and consider a two-sided ideal $I_T \subset A$. Let $G_T = \{g_1, \dots, g_r\}$ be a set of generators of I_T. We say that G_T is a **two-sided σ-Gröbner basis** of I_T if it satisfies one of the following three equivalent conditions:

(1) $\langle G_T \rangle_L = \langle G_T \rangle_T = I_T$

(2) $\langle G_T \rangle_R = \langle G_T \rangle_T = I_T$

(3) $\langle G_T \rangle_L = \langle G_T \rangle_R = I_T$

In fact, from ([24],Theorem 5.4), the above equalities (2) and (3) follow from (1).

Remark 6.1.8. If a finite subset G is a left σ-Gröbner basis of the left ideal $\langle G \rangle_L$ and also a right Gröbner basis of the right ideal $\langle G \rangle_R$, then in general $\langle G \rangle_L \neq \langle G \rangle_R$. For instance, consider the Weyl algebra $A_1 = \mathbb{F}_{11}[x, \partial]$ with $\sigma = \texttt{DegRevLex}$, then $G = \{x\}$ is left Gröbner basis of $\langle x \rangle_L$ and is also a right Gröbner basis of $\langle x \rangle_R$. Now, $xy + 1 \in \langle x \rangle_L$, whereas $xy + 1 \notin \langle x \rangle_R$. This implies that $\langle x \rangle_L \neq \langle x \rangle_R$. Therefore, G is not a two-sided Gröbner basis of $\langle x \rangle_T$, the two-sided ideal generated by $\{x\}$. In fact, $\langle x \rangle_T$ is not proper.

We are now ready to present an algorithm for computing a two-sided Gröbner basis of a two-sided ideal $I_T \subset A_n$. As stated above, the algorithm works as follows.

Algorithm 6.1.9. *Two-sided Gröbner Basis Algorithm*: $\texttt{TwoWGB}(I_T)$

Let I_T be a two-sided ideal of Weyl algebra A_n of index n over a field $K = \mathbb{F}_p$.

`Input:` Ideal $I_T := \langle f_1, \dots, f_s \rangle$ of A_n and a term ordering σ.

`Output:` A two-sided Gröbner basis for I_T with respect to σ

Perform the following sequence of steps.

(1) Compute a left σ-Gröbner basis G_L of I_T.

(2) Multiply every element of L form the right side with the $2n$ indeterminates of A_n selecting one at a time.

(3) If the normal remainder of the above product with respect to G_L is non-zero then add it to the set G_L.

(4) After performing Steps (2) and (3) for each indeterminate, stop if G_L is not changed. Otherwise replace G_L by a left Gröbner basis of the ideal generated by G_L and continue with Step (2).

Proposition 6.1.10. *Algorithm 6.1.9 terminates and returns a two-sided Gröbner basis of the ideal I_T with respect to the term ordering σ.*

Proof. For the proof we refer to [30] (Algorithm 3.1). □

The following observation will be important for constructing instances of cryptosystems

Proposition 6.1.11. *Let $I_T = \langle f_1, \dots, f_r \rangle_T$ be a two-sided ideal of the Weyl algebra $A_n = \mathbb{F}_p[x_1, \dots, x_n, \partial_1, \dots, \partial_n]$ and let σ be a term ordering on A_n. If the generating polynomials $f_1, \dots, f_r$ of I_T are contained in the center C_n, then the following claims hold:*

(1) The ideal I_T, viewed as a left (resp. right) ideal of A_n, its left (resp. right) σ-Gröbner basis G_L will be contained in the center C_n.

(2) The two-sided σ-Gröbner basis G_T of I_T is contained in C_n.

Proof. Since for $i = 1, \dots, r$ we have $f_i \in C_n$, therefore, $\mathrm{Supp}(f_i) \subset C_n$. In particular, for each i we have $\mathrm{LT}_\sigma(f_i) \in C_n$. Therefore, for any pair (f_i, f_j), we have $\mathrm{lcm}(\mathrm{LT}_\sigma(f_i), \mathrm{LT}_\sigma(f_j)) \in C_n$ and hence the S-polynomial of f_i and f_j belongs to the center C_n. Since C_n is a commutative polynomial ring, it follows that all the intermediate and final results obtained by the left Division Algorithm 2.3.18 are the elements of C_n. Therefore, the left σ-Gröbner basis G_L obtained as an output of the left Buchberger Algorithm 2.3.24 will be contained in C_n. This completes the proof of (1).

We can now prove part (2). From Part (1), the left σ-Gröbner basis G_L is contained in C_n. Note that in Algorithm 6.1.9, for computing two-sided Gröbner basis G_T, we first compute G_L. Let $\mathscr{G}_L$ be the tuple of polynomials in G_L. Then for

$i = 1, \dots, n$, and for every $g \in G_L$, we have $\mathrm{NR}_{\sigma,\mathscr{G}_L}(g\,x_i) = 0$ and $\mathrm{NR}_{\sigma,\mathscr{G}_L}(g\,\partial_i) = 0$. This follows from the fact that $G_L \subset C_n$ is left Gröbner basis and both $gx_i = x_i g \in I_T$ and $g\partial_i = \partial_i g \in I_T$. Therefore in the Step (3) of Algorithm 6.1.9, nothing will be added to the set G_L. Hence in this case $G_T = G_L$ and the claim follows. $\square$

We shall now provide some examples of two-sided Gröbner bases of two-sided ideals of A_n.

Example 6.1.12. For the Weyl algebra $A_1 = \mathbb{F}_7[x, \partial]$ with $\sigma = \texttt{DegRevLex}$, consider the subset $S = \{x^7y^7 + 1, xy^2 - 1\} \subset A_1$. Then a two-sided σ-Gröbner basis of the ideal $\langle S \rangle_T$ generated by $S \subset A_1$ turns out to be $G_T = \{1\}$. Hence $\langle S \rangle_T$ is a trivial two-sided ideal of A_1, whereas the reduced left σ-Gröbner basis of the left ideal $\langle S \rangle_L$ is $\{g_1, \dots, g_4\}$ where

$$\begin{aligned}
g_1 &= y^4 - y^3 - x^2 + 2xy - x - 2, \\
g_2 &= x^2y + y^3 + x^2 - 3xy - 3x + 3, \\
g_3 &= x^3 - 3y^3 + 3x^2 + xy - 2y^2 + 3x + y - 1, \\
g_4 &= xy^2 - 1.
\end{aligned}$$

Hence $\langle S \rangle_L$ is a *proper* left ideal of A_1.

Example 6.1.13. Consider the Weyl algebra $A_3 = \mathbb{F}_3[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$ with $\sigma = \texttt{DegRevLex}$. Choose polynomials

$$\begin{aligned}
f_1 &= x_1^6x_2^3\partial_1^6 - x_2^3x_3^3\partial_2^6 + x_3^3\partial_3^6 - \partial_1^3 + \partial_3^3 - 1, \\
f_2 &= x_1^3\partial_1^6 - x_2^3\partial_2^3 + x_3^3\partial_3^3 - x_1^3 + \partial_2^3 - 1
\end{aligned}$$

in A_3 and consider the two-sided ideal $I_T = \langle f_1, f_2 \rangle_T$ generated by these two polynomials. Then the implementation of Algorithm 6.1.9 returns the set $G_T = \{g_1, g_2, g_3\}$ as the reduced two-sided Gröbner basis of the ideal I_T, where

$$\begin{aligned}
g_1 &= x_2^3x_3^3\partial_1^6\partial_2^6 - x_2^9\partial_2^6 - x_2^6x_3^3\partial_2^3\partial_3^3 - x_2^3x_3^6\partial_3^6 - x_3^3\partial_1^6\partial_3^6 - x_2^6\partial_2^6 + x_2^3x_3^3\partial_2^6 + \\
&\quad x_2^3x_3^3\partial_2^3\partial_3^3 + x_1^6x_2^3 + \partial_1^9 + x_2^6\partial_2^3 - x_2^3\partial_2^6 - x_2^3x_3^3\partial_3^3 - \partial_1^6\partial_3^3 - x_3^3\partial_3^6 + \partial_1^6 - \\
&\quad x_2^3\partial_2^3 - x_2^3 + \partial_1^3 - \partial_3^3 + 1, \\
g_2 &= x_1^3x_2^6\partial_2^3 - x_2^3x_3^3\partial_2^6 - x_1^3x_2^3x_3^3\partial_3^3 + x_1^6x_2^3 - x_1^3x_2^3\partial_2^3 + x_3^3\partial_3^6 + x_1^3x_2^3 - \partial_1^3 + \partial_3^3 - 1, \\
g_3 &= x_1^3\partial_1^6 - x_2^3\partial_2^3 + x_3^3\partial_3^3 - x_1^3 + \partial_2^3 - 1.
\end{aligned}$$

Note here that G_T is also a left σ-Gröbner basis of the left ideal $I_T = \langle f_1, f_2\rangle_T$. Moreover, $G_T \subset C_n = \mathbb{F}_3[x_1^3, x_2^3, x_3^3, \partial_1^3, \partial_2^3, \partial_3^3]$.

6.2 Two-sided Weyl Gröbner Basis Cryptosystems

Keeping in mind the properties and the structure of two-sided ideals in Weyl algebras, we are now ready to introduce two-sided Weyl Gröbner Basis Cryptosystems (TWGBC). As before, let the field $K = \mathbb{F}_p$ be a finite field of characteristic p and A_n be the Weyl algebra of index n over K. Let the K-basis B_n of A_n be as given in Equation (6.1) and let σ be a term ordering on B_n. Further recall that, given a set of Weyl polynomials $G = \{g_1, \ldots, g_r\} \subset A_n \setminus \{0\}$, we can use the *left Division Algorithm* to compute the *normal remainder* $\mathrm{NR}_{\sigma,\mathscr{G}}(f)$ of any polynomial $f \in A_n$ with respect to the tuple $\mathscr{G} = (g_1, \ldots, g_r)$ (see Algorithm 2.3.18 and Definition 2.3.20). Moreover, if G_T is a two-sided σ-Gröbner basis of the two-sided ideal I_T, then then it will also be a left σ-Gröbner basis of the left ideal generated by G_T, i.e. $\langle G_T\rangle_L = \langle G_T\rangle_T = I_T$. It turns out that every Weyl polynomial $f \in A_n$ has a unique normal remainder $\mathrm{NR}_{\sigma,\mathscr{G}_T}(f)$ (see Theorem 2.4.1), and that if $f \in I_T$ then $\mathrm{NR}_{\sigma,\mathscr{G}_T}(f) = 0$ (Theorem 2.4.1, Part (2)). With these ingredients, we are now ready to introduce the following cryptosystems.

Cryptosystem 6.2.1. Given a Weyl algebra A_n of index n over $K = \mathbb{F}_p$, let I_T be a non-trivial two-sided ideal of A_n and let $G_T = \{g_1, \ldots, g_r\}$ be its two-sided σ-Gröbner basis. We set $\mathscr{G}_T = (g_1, \ldots, g_r)$ and $\mathscr{O}_\sigma(I_T) = B_n \setminus \{\mathrm{LT}_\sigma(f) \mid f \in I_T \setminus \{0\}\}$. Then a **two-sided Weyl Gröbner basis cryptosystem (TWGBC)** consists of the following data.

(1) **Public Key:** A set Q of Weyl polynomials $\{p_1, \ldots, p_s\}$ contained in $I_T \setminus \{0\}$ and a subset $\mathscr{M}$ of $\mathscr{O}_\sigma(I)$ are known publicly.

(2) **Secret Key:** The reduced two-sided σ-Gröbner basis $G_T = \{g_1, \ldots, g_r\}$ of the ideal I_T and the set $\mathscr{O}_\sigma(I_T)$ are kept secret.

(3) **Message Space:** The message space is the K-vector subspace $\langle \mathscr{M}\rangle_K$ of A_n generated by $\mathscr{M} \subset \mathscr{O}_\sigma(I_T)$.

(4) **Ciphertext Space:** The ciphertext units are Weyl polynomials in A_n.

(5) **Encryption:** For encrypting a plaintext message $m \in \langle \mathscr{M} \rangle_K$, choose Weyl polynomials ℓ_i and r_i, and then compute the standard form of

$$c = \sum_{i=1}^{s'} \ell_i \, p_{k_i} r_i, \text{ where } s' \geq s \text{ and } k_i \in \{1, ..., s\},$$

to get the ciphertext polynomial $c \in A_n$.

(6) **Decryption:** Given a ciphertext polynomial $c \in A_n$, compute $\mathrm{NR}_{\sigma,\mathscr{G}_T}(c)$. If the result is contained in $\langle \mathscr{M} \rangle_K$, return it. Otherwise, return c.

Note here that since G_T is a two-sided σ-Gröbner basis of the ideal I_T and the polynomials $p_1, \ldots, p_s \in I_T$, it follows that we have $\mathrm{NR}_{\sigma,\mathscr{G}_T}(p_i) = 0$ for each $i = 1, \ldots, s$, (see Theorem 2.4.1.2). This implies that for $k_i \in \{1, \ldots, s\}$

$$\mathrm{NR}_{\sigma,\mathscr{G}_T}(m + \sum_i \ell_i \, p_{k_i} r_i) = m,$$

and hence the correctness of the system follows.

Note. From now onwards, we abbreviate a two-sided Weyl Gröbner basis cryptosystem as TWGBC.

Again the security of TWGBC strongly depends on the difficulty of computing two-sided Gröbner basis in Weyl algebras. That is, if an attacker can compute G_T, he can break the cryptosystem. Together with the subset of $\mathscr{O}_\sigma(I)$ the attacker only knows the Weyl polynomials $\{p_1, \ldots, p_s\}$ in the public key $Q \subset I_T$. Therefore, they have to be created in a way that hides all information about the system of generators of I_T. In particular, the leading terms of polynomials in the secret key should be well hidden. On the other hand, the attacker might also try to compute a two-sided σ-Gröbner basis of the ideal $J_T = \langle Q \rangle_T$ generated by the set of polynomials in the public key. But, in the setting of Weyl algebras, as in the case of WGBC (see Section 4.1), we can make this task difficult by choosing suitable polynomials in the public key $Q = \{p_1, \ldots, p_s\}$ such that a two-sided σ-Gröbner basis of the ideal $J_T = \langle p_1, \ldots, p_s \rangle_T$ is hard to compute. To show the existence of such ideals in Weyl algebras, we present an easily construct example below.

Example 6.2.2. Let the Weyl algebra A_n, the term ordering σ and the two-sided ideal $I_T \subset A_n$ be as given in Example 6.1.13. Then a two-sided Gröbner basis of this ideal is the set $G_T = \{g_1, g_2, g_3\}$, as given in the same example. We choose two random sparse polynomials $p_1, p_2 \in I_T$ such that $\deg(p_1) = 18$ and $\deg(p_2) = 17$. The number of terms in the standard form of the polynomials p_1 and p_2 are 204 and 198 respectively. It is very easy and straightforward to choose such polynomials in the ideal I_T by using any computer algebra system. For instance, if f is a dense polynomial in A_n such that $\deg(f) = 18$ then $\mathrm{Supp}(f)$ can contain at most 134596 terms. For getting a sparse polynomial in A_3, we first randomly choose less than one percent i.e. between 1000 - 1300 terms in the $\mathrm{Supp}(f)$ and randomly assign them coefficients from $K = \mathbb{F}_3$ to obtain a new random-looking sparse polynomial $f' \in A_3$. Now we can set $p_1 = f' - \mathrm{NR}_{G_T}(f')$ and get another random-looking polynomial $p_1 \in I_T$. The polynomials p_1, and p_2 are given in Appendix C.3. Now consider the set $Q = \{p_1, p_2\}$ and let $J_T = \langle Q \rangle_T$ be the two-sided ideal generated by Q. Then there is a significant computational evidence that a two-sided Gröbner basis of the ideal J_T is hard to compute. In this case, using the CAS `Singular`, our computing machine failed to compute not only a two-sided Gröbner basis but also the computation of a left Gröbner basis of the ideal $\langle Q \rangle_L$ was found to be infeasible. This claim is based on the observation that our computation has consumed more than 3 GB of memory when we stopped it after 38,422.8 seconds of CPU time. At the time of interruption, computations were progressing too slow due to very large size of the resulting polynomials.

Remark 6.2.3. It is remarkable to point out here that in the above example and many other similar cases, it is the very slow reduction process that makes the computation of two-sided Gröbner basis of the ideal $J = \langle Q \rangle_T$ infeasible. After couple of hours of computation, the sizes of the resulting intermediate Weyl polynomials grow too large to compute their normal remainder effectively.

From these computational results, we claim that it is easy to construct a public key Q for a TWGBC such that a two-sided Gröbner basis of the ideal $J = \langle Q \rangle_T$ is hard to compute. This claim is based on the results obtained by using an implementation of Algorithm 6.1.9 for computing two-sided Gröbner bases of ideals in Weyl algebras. But this is not sufficient for constructing a secure instance of TWGBC.

Rather, one also has to make sure that various attacks proposed by the cryptanalysts of the Gröbner basis type cryptosystems are either not applicable or are not practical in the setting of TWGBC. As in the case of WGBC, we can achieve this objective by fixing parameters of our proposed TWGBC and the way of choosing public polynomials and various other Weyl polynomials required for the encryption process. In the following remark, let us first observe an important advantage of using a two-sided Weyl Gröbner basis cryptosystem.

Remark 6.2.4. In the encryption process the ciphertext polynomial c is computed as

$$c = \sum_{i=1}^{s'} \ell_i \, p_{k_i} \, r_i, \text{ where } s' \geq s \text{ and } k_i \in \{1, ..., s\}.$$

Note that for computing c, the sender *Alice* needs two sets of polynomials, namely the polynomials $\ell_1, \ldots, \ell_{s'}$ and the polynomials $r_1, \ldots, r_{s'}$. That is, for each p_{k_i} she needs a polynomial ℓ_i for the left multiplication and a polynomial r_i for the multiplication from the right-hand side with p_{k_i}. Hence one obvious advantage of using a TWGBC over a WGBC is that the TWGBC is not vulnerable to the very serious attacks based on linear algebra of Section 5.2. In this setting, the resulting polynomial system of equations will be quadratic. Such systems are much harder to solve than systems of linear equations.

The hardness of solving the above mentioned system of equations also depends on the various parameters of a TWGBC. These parameters are same as the parameters given in Notation 4.1.5 for WGBC, except for one additional parameter d_r, the maximum degree of the polynomials $r_1, \ldots, r_{s'}$ used for the encryption. Moreover, unlike WGBC, for TWGBC the field characteristic has to be positive which is obviously needed for the existence of two-sided ideals of a Weyl algebra A_n.

In the next section, we shall now provide a procedure for the key generation and implementation of practical instances of TWGBC.

6.3 TWGBC Key Generation and Implementation

In the following Procedure 6.3.1 we introduce a step-by-step method for generating a pair (G, Q) for constructing concrete instances of TWGBC. That is, by following these steps, one can generate an apparently secure secret key and a presumably hard to break ciphertext.

Procedure 6.3.1. Let A_n be a Weyl algebra of index n over the field $K = \mathbb{F}_p$ and let B_n be its set of terms. Let σ be a term ordering on B_n. Then, to construct a concrete hard instance of Cryptosystem 6.2.1, perform the following steps.

(1) Choose a non-trivial two-sided ideal I_T of A_n such that its two-sided Gröbner basis is easy to compute. Let $G_T = \{g_1, \dots, g_r\}$ be the reduced two-sided Gröbner basis of the ideal I_T such that $G_T \subset C_n$. Let $d_g = \max\{\deg(g) \mid g \in G_T\}$.

(2) For $i = 1, \dots, s$ choose random sparse polynomials $p_i \in I_T$ of sufficient high degree as compare to the degree d_g. This can be done for instance by following (2a) or (2b) below:

(2a) Choose random sparse polynomials $f'_1, \dots, f'_q \in A_n$ of degrees greater than d_g. For $i = 1, \dots, q$, compute $f_i = f'_i - \mathrm{NR}_{\sigma,\mathscr{G}_T}(f'_i)$. Then, for each i, $f_i \in I_T$, and $\mathrm{Supp}(f_i)$ will also contain terms that are not contained in the center C_n. Keeping these polynomials secret, choose the polynomials h_{ij} and s_{ij} in A_n and compute the standard form of the Weyl polynomials

$$p_i = h_{i1}\, f_1\, s_{i1} + \cdots + h_{iq}\, f_q\, s_{iq}.$$

While choosing the polynomials h_{ij} and s_{ij}, make sure that the degree forms $\mathrm{DF}(h_{ij} f_j s_{ij})$ cancel. The other degree terms of $h_{ij}\, f_j\, s_{ij}$ cancel or their coefficients are changed in p_i by the process of converting the remaining $h_{ik}\, f_k\, s_{ik}$ to standard form. In this way, no important information about the polynomials in the secret key G_T should be visible in p_i.

(2b) Since, $g_1, \dots, g_r \in C_n$, for $i = 1, \dots, s$ and $j = 1, \dots, r$, choose the polynomials $h_{ij} \in A_n$, and compute the standard form of the Weyl polynomials

$$p_i = h_{i1}\, g_1 + \cdots + h_{ir}\, g_r.$$

While choosing the polynomials h_{ij}, make sure that the degree forms $\mathrm{DF}(h_{ij} g_j)$ of highest degree cancel.

Let the set $Q = \{p_1, \ldots, p_s\}$ be the public key.

(3) Let $J_T = \langle p_1, \ldots, p_s \rangle_T$ be the two-sided ideal generated by the polynomials in the public key Q. Make sure that not only the complete two-sided σ-Gröbner basis of the ideal J_T is hard to compute, but also a *partial Gröbner basis* is infeasible to compute for large degree bounds.

(4) Choose a subset $\mathscr{M} \subset \mathscr{O}_\sigma(I_T)$ for the message space $\langle \mathscr{M} \rangle_K$ in such a way that every g_i contains at least one term in $\mathscr{O}_\sigma(I_T) \setminus \mathscr{M}$.

(5) For constructing a ciphertext polynomial

$$c = \sum_{i=1}^{s'} \ell_i \, p_{k_i} r_i, \quad \text{where } s' \geq s \text{ and } k_i \in \{1, ..., s\},$$

choose the polynomials $\ell_1, \ldots, \ell_{s'}$ and $r_1, \ldots, r_{s'}$ such that the following properties hold:

(a) Make sure that $\mathrm{Supp}(\sum_{i=1}^{s'} \ell_i \, p_{k_i} r_i)$ contains all terms of $\mathrm{Supp}(m)$ and many terms of $\mathscr{M}$. In this way, the monomials of m will be either cancelled or their coefficients will be changed in the lower degree part of the polynomial c.

(b) Ascertain that the degree forms $\mathrm{DF}(\ell_i p_{k_i} r_i)$ cancel in c, and that the other degree forms $\mathrm{DF}(\ell_i p_{k_i} r_i)$ cancel or their coefficients are changed in c by the process of converting the remaining $\ell_j p_{k_j} r_j$ to standard form.

(c) Again, in meeting properties (a) and (b) above, use sufficiently high powers of $\partial_1, \ldots \partial_n$ in the terms of the support of ℓ_i and high powers of $x_1, \ldots, x_n$ in the terms of the support of r_i such that, after bringing $\ell_i \, p_{k_i} r_i$ to standard form, there are no wide gaps in degrees of various terms in $\mathrm{Supp}(c)$. This means that due to expansion of the ciphertext polynomial during Weyl multiplication, the sparsity of the polynomial c will be reduced and it will be more 'random-looking'.

(6) Make sure that with the above choices of the polynomials $\ell_1, \ldots, \ell_{s'}$ and $r_1, \ldots, r_{s'}$, the degree, d_c, of the ciphertext c becomes high enough such that no partial two-sided Gröbner basis of the ideal J_T can be computed for large

degree bounds. Moreover, if $\mathscr{H}$ is a partial Gröbner basis of J_T for a degree bound less than d_c, then ensure that $\mathrm{NR}_{\sigma,\mathscr{H}}(c) \neq m$.

In Section 6.4, we shall see that, if we follow the the steps of Procedure 6.3.1, the standard attacks become infeasible. In fact, step (2) makes sure that the polynomials in the secret key G_T are well concealed. The step (5) ensures that not only the plaintext message m is well hidden in the ciphertext polynomial c, but, by reducing the sparsity of the polynomial c and by removing gaps in the degrees of the terms in the support of c, we are, making c more 'random-looking'. Similarly, by completing the steps (3) and (4), we are, respectively making the *partial Gröbner basis attack* and the *chosen ciphertext attack* infeasible (see Section 6.4 for details).

Let us now try to construct a concrete instance of a TWGBC. In the following example, we follow Step (2b) for creating a public key Q.

Example 6.3.2. Consider the Weyl algebra $A_2 = \mathbf{Z}_{13}[x_1, x_2, \partial_1, \partial_2]$ and let the term ordering be $\sigma = \mathrm{DegRevLex}$. Choose a subset $\{F_1, F_2\} \subset A_n$ where

$$F_1 = x_1^{13} x_2^{26} \partial_1^{26} - 2 \text{ and } F_2 = 3x_2^{26} + 2x_2^{13},$$

Let $I_T = \langle \{F_1, F_2\} \rangle_T$ be the two-sided ideal generated by this subset. then the reduced two-sided Gröbner basis of I_T is the set $G_T = \{g_1, g_2\}$, where

$$g_1 = x_2^{13} + 5 \text{ and } g_2 = x_1^{13} \partial_1^{26} + 2.$$

We now introduce the following TWGBC

(1) **Secret Key:**
The secret key is the two-sided Gröbner basis $G_T = \{g_1, g_2\}$. Let $\mathscr{G}_T = (g_1, g_2)$.

(2) **Public Key:**

Choose

$$\begin{aligned} f_1' &= x_2^{13}\partial_2 + 3x_2^{14} + 5x_2^{13} - 2x_1^{13}\partial_1^{26}x_2^3 + 2x_2^{13}\partial_2 + 3x_2^{13}\partial_1\partial_2 - x_1^{14}\partial_1^{28}\partial_2^2 - \\ &\quad x_2^{13}\partial_1^2 - x_1^3x_2\partial_2^3 - x_2^2\partial_2 - 7 \\ f_2' &= 2x_2^{13}\partial_2^2 - 3x_2^{13}\partial_1^2 + x_1^{13}x_2^{26}\partial_1^{26} - 3x_1^3x_2^{15}\partial_2^2 + 4x_1^{14}x_2^2\partial_1^{28} - 2x_2^{13}\partial_2^{13} + \\ &\quad x_2^{13}\partial_1\partial_2^2 - 3\partial_2^{11}x_2^{10} \end{aligned}$$

and compute $f_1 = f_1' - \mathrm{NF}_{\sigma,G_T}(f_1')$ and $f_2 = f_2' - \mathrm{NF}_{\sigma,G_T}(f_2')$. Then

$$\begin{aligned} f_1 &= -x_1^{14}\partial_1^{28}\partial_2^2 - 2x_1^{13}x_2^3\partial_1^{26} - x_2^{13}\partial_1^2 + 3x_2^{13}\partial_1\partial_2 + 3x_2^{14} + 3x_2^{13}\partial_2 + 5x_2^{13} \\ &\quad -2x_1\partial_1^2\partial_2^2 - 4x_2^3 - 5\partial_1^2 + 2\partial_1\partial_2 + 2x_2 + 2\partial_2 - 1 \\ f_2 &= x_1^{13}x_2^{26}\partial_1^{26} + 4x_1^{14}x_2^2\partial_1^{28} - 2x_2^{13}\partial_2^{13} - 3x_1^3x_2^{15}\partial_2^2 + x_2^{13}\partial_1\partial_2^2 - 3x_2^{13}\partial_1^2 + \\ &\quad 2x_2^{13}\partial_2^2 + 3\partial_2^{13} - 2x_1^3x_2^2\partial_2^2 - 5x_1x_2^2\partial_1^2 + 5\partial_1\partial_2^2 - 2\partial_1^2 - 3\partial_2^2 - 2 \end{aligned}$$

Using f_1 and f_2, we can create polynomials $p_1, p_2, \ldots$ for the public key Q by computing the standard forms of

$$\begin{aligned} p_1 &= h_{11}f_1s_{11} + h_{12}f_2s_{12}, \\ p_2 &= h_{21}f_1s_{21} + h_{22}f_2s_{22}. \end{aligned}$$

Here we let

$$\begin{aligned} h_{11} &= x_2^{16} + 3\partial_1 + 2\partial_2^3 - 1, & s_{11} &= x_2^{10} + 3x_1^3 - 1, \\ h_{12} &= x_1\partial_1\partial_2^2, & s_{12} &= \partial_1, \\ h_{21} &= x_2^8\partial_2 + x_2^{12}, & s_{21} &= x_2^{20}\partial_2 + x_2^{11}, \\ h_{22} &= x_1\partial_1\partial_2^2 + 5\partial_1\partial_2^2 + 2, \text{ and} & s_{22} &= x_2^2\partial_1\partial_2^2 + 3x_1^2 - 2x_2^2 + 1. \end{aligned}$$

Then the Weyl polynomial p_1 has degree 68 and its standard form consists of 332 terms. The Weyl polynomial p_2 has degree 77 and there are 531 terms in its standard form. These polynomials p_1 and p_2 are given in Appendix C.3.

(3) **Message Space:**

For the message space, we choose the K-vector space generated by the set

$$\mathscr{M} = \{x^\alpha \partial^\beta \mid |\alpha| \le 9,\ |\beta| \le 10\}.$$

There are 13^{3660} different possible plaintext units.

(4) **Encryption:**

To encrypt a message $m \in \langle \mathscr{M} \rangle_K$, we use Step (5) of Procedure 6.3.1 and choose polynomials $\ell_1, \ell_2, \ell'_1, \ell'_2$ and r_1, r_2, r'_1, r'_2 of sufficiently high degree and compute the standard form of the ciphertext polynomial

$$c = \sum_{i=1}^{s'} \ell_i \, p_{k_i} \, r_i, \text{ where } s' \geq 2 \text{ and } k_i \in \{1, 2s\},$$

For instance, to encrypt a message

$m = -3x_1^2\partial_1^4\partial_2^4 + 6x_1^2x_2^3\partial_2^5 - x_1x_2\partial_1^3\partial_2^5 + 3x_1^3x_2^3\partial_1^3 - 2x_1^3\partial_1^6 + 4x_1^2\partial_1^7 - 2x_1^6x_2\partial_1\partial_2 - x_1^7\partial_2^2 + x_1^2x_2^3\partial_1^2\partial_2^2 + 3x_1^3\partial_1^4\partial_2^2 + 3x_1x_2\partial_1^5\partial_2^2 + 5x_2\partial_1^6\partial_2^2 - 4x_1^4\partial_1^2\partial_2^3 + 6x_2^4\partial_2^5 + 3x_1\partial_2^8 + 3x_1^2x_2^6 - 3x_1^2x_2\partial_1^4\partial_2 + 6x_1^4\partial_1^2\partial_2^2 + x_1x_2\partial_1^4\partial_2^2 - 6x_1x_2^2\partial_1^2\partial_2^3 + 3\partial_1^5\partial_2^3 + 3\partial_1^2\partial_2^6 + 4x_2\partial_2^7 + x_1^2x_2^5 + 2x_1^3\partial_1^4 - 4x_1x_2\partial_1^5 + 2x_1^2x_2^4\partial_2 + 6x_1x_2^5\partial_2 - 2x_1^2x_2\partial_1^3\partial_2 - 2x_1^4\partial_1\partial_2^2 + x_1^3x_2\partial_1\partial_2^2 + 6x_1^2x_2\partial_2^4 - 3\partial_1^2\partial_2^5 + 2x_1x_2^2\partial_2^3 + 6x_1^3x_2^2 - 2x_1x_2^3\partial_1 - 6\partial_1^4 + 5x_1^2\partial_2^2 + \partial_1$,

we may choose

$$\begin{aligned}
\ell_1 &= 3x_1^4\partial_1^6\partial_2^6 + x_1x_2^2\partial_1^3\partial_2^2, & r_1 &= x_1^2x_2\partial_1\partial_2^3,\\
\ell_2 &= -4x_1^2\partial_1^4\partial_2^3 + x_1\partial_1^2 - x_1\partial_1\partial_2 + \partial_1, & r_2 &= x_1^2\partial_1^2\partial_2^3 - x_1\partial_1^2 + x_1\partial_1\partial_2 - x_1,\\
\ell_3 &= -x_1^2x_2^2, & \ell_4 &= \ell_5 = r_3 = 1,\\
r_4 &= x_1^4x_2^6\partial_1^6\partial_2^7 - 3x_1^6x_2\partial_1^7\partial_2^9, & r_5 &= 4x_1^4\partial_1^6\partial_2^6 + 6x_1^3\partial_1^5\partial_2^6.
\end{aligned}$$

and compute the standard form of

$$c = m + \ell_1 \, p_1 \, r_1 + \ell_2 \, p_2 \, r_2 + \ell_3 \, p_1 \, r_3 + \ell_4 \, p_1 \, r_4 + \ell_5 \, p_2 \, r_5.$$

In the above representation of c we obtain a ciphertext polynomial of degree 89 and its standard form consists of 13,175 terms. The polynomials ℓ_i, r_i are chosen such that the highest degree form of the ciphertext polynomial c cancels. For instance, we have $\deg(p_1) = 68$ and $\mathrm{LT}_\sigma(p_1) = 6x_1^{14}x_2^{25}\partial_1^{28}\partial_2$. We choose a random term $t_{\ell_1} = 3x_1^4\partial_1^6\partial_2^6$ of degree 16 for ℓ_1 and another random term $t_{r_1} = x_1^2x_2\partial_1\partial_2^3$ of degree 7 for r_1. Now the degree of the product $t_{\ell_1} \cdot p_1 \cdot t_{r_1}$ is 91 and its leading term is $5x_1^{20}x_2^{26}\partial_1^{35}\partial_2^{10}$, to cancel it from c, choose $-3x_1^6x_2\partial_1^7\partial_2^9$ of degree 23 as a term in r_4. This cancels the above leading term of degree 91 from c. Now choose another term $t_{r_4} = x_1^4x_2^6\partial_1^6\partial_2^7$ for r_4, then the leading term of the product $p_1 \, r_4$ is $6x_1^{18}x_2^{31}\partial_1^{34}\partial_2^8$ and its degree is again 91. Again to cancel it from c, we choose terms in ℓ_2, r_2 and r_5

such that $-2x_1^{18}x_2^{31}\partial_1^{34}\partial_2^{8}$ appears in the product $\ell_2\, p_2\, r_2$ and $-4x_1^{18}x_2^{31}\partial_1^{34}\partial_2^{8}$ appears in the product $p_2\, r_5$ and this cancels $6x_1^{18}x_2^{31}\partial_1^{34}\partial_2^{8}$ in c. Note also that for the term $t_{r_4} = 4x_1^4\partial_1^6\partial_2^6$ chosen for r_4 and setting $\ell_4 = 1$, we can cancel many terms in the product $\ell_4\, p_2\, r_4$ by using various possible factors of t_{r_4} for the left and the right multiplication with p_2. For instance, among many possibilities, we choose a term $t_{\ell_2} = -4x_1^2\partial_1^4\partial_2^3$ for ℓ_2 and the corresponding factor $t_{r_2} = x_1^2\partial_1^2\partial_2^3$ for r_2. Note the strategy of choosing the terms t_{ℓ_2} and t_{r_2} such that $t_{\ell_2} * t_{r_2}$ becomes equal to t_{r_4}, here $*$ means the multiplication in the commutative sense. This does not only cancel the leading term of $1 \cdot\ p_2 \cdot t_{r_4}$ in c but altogether 531 terms are cancelled in the sum $t_{\ell_2} \cdot p_2 \cdot t_{r_2} + 1 \cdot p_2 \cdot t_{r_4}$. All the terms that are left in this sum are due to Weyl multiplication. Continuing this way, we keep on adding and setting various terms for ℓ_i and r_i and finally compute c as above. In this way, many terms in c are either cancelled or their coefficients are changed. The degree form $\mathrm{DF}(c)$ contains 7 terms of degree $d_c = 89$. This means that all the terms of degree greater than 89 are cancelled in c. Further, instead of 19, we only have 7 terms of degree 89, i.e. some of the terms of degree 89 are cancelled or their coefficients are changed in c. This can be easily seen by observing the number of terms in the homogeneous components of $\ell_i\, p_{k_i}\, r_i$,for each i and comparing them with the number of terms of the homogeneous components of c.

Moreover, out of 39 monomials of m, 25 are not present in c, and the remaining 14 monomials are mixed in 540 monomials of c from the message space. Therefore, the message m is well-hidden.

(5) **Decryption:**
Since $m = \mathrm{NR}_{\sigma,\mathscr{G}_T}(c)$, to decipher c, it suffices to compute the normal remainder of the ciphertext polynomial c with respect to the secret key $\mathscr{G}_T$. In the present case, the decryption takes 0.79 seconds on our computing machine using the package `Weyl` of ApCoCoA.

Note that in the above Example 6.3.2, not all the requirements of Procedure 6.3.1 are satisfied. For instance, the polynomials g_1 and g_2 of the public key G_T are binomials. Moreover, none of the polynomials g_1 and g_2 have terms from $\mathscr{O}_\sigma(I) \setminus \mathscr{M}$ in their support. If an attacker can guess the leading terms of these polynomials with respect to the term ordering σ, he can try to break the system by using the *chosen ciphertext attack* as described in Section 5.4 for the case of WGBC.

Remark 6.3.3. In the case of a TWGBC, for encryption we need two sets of polynomials, namely the polynomials $\ell_1, \dots, \ell_{s'}$ that are multiplied from the left with each p_{k_i} and the polynomials $r_1, \dots, r_{s'}$ for multiplication from the right. In view of the requirement (6)-(c) the ciphertext polynomial may expand too much and may result in a bad data-rate for transmitting the ciphertext c over a network. For instance, in the above example, the resulting ciphertext contains $13,175$ terms. This gives us a data-rate of approx. 1/337 for transmitting c. To overcome this problem, we suggest to use a message space $\mathscr{M}$ that allows us to represent a plaintext message m with a polynomial of large size. In the above example, considering the size of the message space, message expansion is rather moderate. The message expansion can also be controlled by working in fields with small characteristic such as $\mathbb{F}_2, \mathbb{F}_3, \mathbb{F}_5$, or $\mathbb{F}_7$.

Keeping these observations in mind, we now present a procedure for constructing concrete instances of TWGBC.

6.4 Concrete Hard Instances

As in the case of WGBC, the structure and properties of Weyl algebras turn out to be very useful in satisfying the requirements of Procedure 6.3.1 for constructing concrete hard instances of TWGBC. In view of Example 6.3.2 and related observations, below we present a procedure that provides an explicit suggestion on how this can be done. The idea is to choose a proper two-sided ideal $I_T \subset A_n$ such that it satisfies condition of Proposition 6.1.11 (see Examples 6.1.12 and 6.1.13).

Procedure 6.4.1. Let $K = \mathbb{F}_p$ be a finite field of characteristic p. Let $n > 2$, and consider the Weyl algebra A_n of index n over K. Let σ be a term ordering on B_n. Then the following instructions define a TWGBC which satisfies Conditions (1) – (6) of Procedure 6.3.1.

(1) For $2 < k \leq n$, choose a (random) set $F = \{f_1, \dots, f_k\}$ of Weyl polynomials such that $F \subset C_n \setminus \mathbb{F}_p$. Moreover, for $i = 1, \dots, k$, every polynomial $f_i \in F$ should be such that

(a) $\deg(f_i) \geq 2p$

(b) The number of terms in support of each f_i should be at least 3. This will be helpful in satisfying requirement (2) below.

Let $I_T = \langle F \rangle_T$ be the two-sided ideal generated by F. Then by Proposition 6.1.11, a two-sided σ-Gröbner basis G_T will be a subset of C_n and hence I_T is a non trivial two-sided ideal in A_n. Moreover, it will be very likely that for every polynomial $g \in G_T$, we will have $\deg(g) \geq 2p$ and $\#\mathrm{Supp}(g) \geq 3$.

(2) For the message space, choose the set $\mathscr{M} \subseteq \mathscr{O}_\sigma(I)$ such that every g_i has at least one term from $\mathscr{O}_\sigma(I) \setminus \mathscr{M}$ in its support.

(3) Since every $g_i \in C_n$, create Weyl polynomials $p_1, \ldots, p_s$ of the form

$$p_i = h_{i1}\, g_1 + \cdots + h_{ir}\, g_r$$

by choosing Weyl polynomials $h_{i1}, \ldots, h_{ir} \in A_n$ such that Condition (2b) of Procedure 6.3.1 is satisfied. At this point, we also suggest not to using a polynomial $g \in G_T$ in the construction of more than one polynomial of the public key Q. That is, if there are 6 polynomials $g_1, \ldots, g_6$ in the secret key G_T, then one may use g_1, g_3, g_6 for computing p_1, and g_2, g_4, g_5 for computing p_2. This might be helpful in concealing the secret key well to make it difficult for an attacker to guess it from the public information.

(4) To make the polynomial p_i random-looking and to reduce its sparsity, choose some polynomials h'_i, $q'_i \in A$ and compute the standard form of $p'_i = h'_i\, p_i\, q'_i$. In this way, some other other terms of $h_{ij}\, g_j$ either cancel or their coefficients are changed in p'_i by the process of converting $h'_i\, p_i\, q'_i$ to standard form. Replace p_i by p'_i and set $Q = \{p_1, \ldots, p_s\}$ as the public key. It is an optional step that can be performed after step (3) if it seems that the secret polynomials used for constructing p_i are not well-hidden. Make sure that the size of the support of p_i does not grow too large after performing this step.

Later we will see that by following these steps, we can create a pair (G_T, Q), for a secret communication by using a TWGBC.

Remark 6.4.2. It is interesting to remark here that by construction, the secret key G_T is contained in the center C_n. Therefore, in the decryption process, while computing the normal remainder with respect G_T, the intermediate results will not grow due to Weyl multiplication. This fact can make the decryption process of TWGBC faster as compare to the decryption in WGBC.

Let us now use the instructions of Procedure 6.4.1 to formulate some concrete cases of TWGBC.

Example 6.4.3. Let $n = 3$ and consider the Weyl algebra

$$A_3 = \mathbb{F}_2[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$$

over the field of characteristic 2. Let the term ordering on B_n be $\sigma = \texttt{DegRevLex}$. We now introduce the following TWGBC:

(1) **Secret Key:**
Choose the following polynomials of A_3

$$\begin{aligned}
f_1 &= x_1^6x_2^4 + x_1^4x_2^2 + x_1^2 + 1, & f_2 &= x_2^6 + x_2^4x_3^2 + x_2^2 + 1,\\
f_3 &= \partial_1^6\partial_2^4 + \partial_1^4\partial_2^2 + \partial_1^2 + 1, & f_4 &= \partial_3^8 + x_1^2\partial_2^2\partial_3^2 + 1,\\
f_5 &= x_2^2x_3^{10} + x_3^6 + x_1^2x_3^2 + x_3^2 + 1.
\end{aligned}$$

Let $I_T = \langle f_1, f_2, f_3 f_4, f_5\rangle$ be the two-sided ideal generated by these polynomials. Then the reduced two-sided σ-Gröbner basis G_T of I_T is the set $\{g_1, \ldots, g_{10}\}$ where

$$\begin{aligned}
g_1 &= x_1^4x_3^{10} + x_1^6x_3^6 + x_1^2x_3^{10} + x_1^2x_2^4x_3^4 + x_1^2x_3^8 + x_3^{10} + x_1^6x_3^2 + x_1^4x_2^2x_3^2 + x_1^4x_3^4 +\\
&\quad x_2^4x_3^4 + x_3^8 + x_1^6 + x_2^4x_3^2 + x_1^2x_3^4 + x_2^2x_3^4 + x_1^2x_2^2 + x_3^4 + x_2^2 + x_3^2,\\
g_2 &= x_3^{14} + x_1^2x_3^{10} + x_1^6x_3^2 + x_1^2x_2^4x_3^2 + x_1^2x_2^2x_3^4 + x_2^4x_3^4 + x_3^8 + x_1^2x_2^4 + x_1^2x_2^2x_3^2 +\\
&\quad x_2^4 + x_1^2x_3^2 + x_2^2x_3^2 + x_3^4 + x_1^2 + x_2^2 + x_3^2 + 1,\\
g_3 &= x_2^4x_3^6 + x_2^2x_3^8 + x_3^{10} + x_1^2x_2^4x_3^2 + x_1^2x_2^2x_3^4 + x_2^4x_3^2 + x_2^2x_3^4 + x_3^6 + x_2^4 +\\
&\quad x_1^2x_3^2 + x_2^2x_3^2 + x_3^2 + 1,\\
g_4 &= x_1^8 + x_1^2x_2^4x_3^2 + x_1^2x_2^2x_3^4 + x_2^4x_3^2 + x_2^2x_3^4 + x_2^4 + x_1^2x_3^2 + x_2^2x_3^2 + x_1^2 + x_2^2,
\end{aligned}$$

$$\begin{aligned}
g_5 &= x_1^4x_2^4+x_1^4x_2^2x_3^2+x_1^6+x_1^2x_2^4+x_1^2x_2^2x_3^2+x_1^4+x_1^2x_2^2+x_2^4+x_1^2x_3^2+x_2^2x_3^2+\\
&\quad x_1^2+x_2^2+x_3^2+1,\\
g_6 &= x_2^2x_3^{10}+x_3^6+x_1^2x_3^2+x_3^2+1,\\
g_7 &= x_1^6x_2^2+x_1^2x_2^4+x_1^2x_2^2x_3^2+x_1^4+x_2^4+x_2^2x_3^2+x_1^2+1,\\
g_8 &= \partial_1^6\partial_2^4+\partial_1^4\partial_2^2+\partial_1^2+1,\\
g_9 &= \partial_3^8+x_1^2\partial_2^2\partial_3^2+1\\
g_{10} &= x_2^6+x_2^4x_3^2+x_2^2+1.
\end{aligned}$$

The set G_T is our secret key. Moreover, the set $\mathscr{O}_\sigma(I_T)$ is also kept secret and only a subset of it will disclosed publicly for the message space.

(2) **Public Key:**
Let us now create public polynomial p_1, p_2, p_3 by choosing

$$\begin{aligned}
h_{11} &= x_1^9x_3^5\partial_1^5\partial_3^3+x_1^9x_3^4\partial_1^5\partial_3^2+x_1^8x_3^5\partial_1^4\partial_3^3+x_1^8x_3^4\partial_1^4\partial_3^2+x_1^8x_3^3\partial_1^2\partial_3^2+\\
&\quad x_1^6x_3^2\partial_1^3\partial_3+x_1^5,\\
h_{12} &= x_1^{13}x_3\partial_1^5\partial_3^3+x_1^{13}\partial_1^5\partial_3^2+x_1^{12}x_3\partial_1^4\partial_3^3,\\
h_{13} &= x_1^3x_3^{13}\partial_1^2\partial_3^2+x_1^3x_3^{12}\partial_1^2\partial_3+x_1^2x_3^{12}\partial_1^3\partial_3+x_1^2x_3^{13}\partial_1\partial_3^2+x_1^2x_3^{12}\partial_1\partial_3+x_1x_3^{10}.
\end{aligned}$$

and then compute the standard form of

$$p_1 = h_{11}\,g_1+h_{12}\,g_2+h_{13}\,g_4.$$

The polynomial p_1 has degree 34 and consists of 222 terms in its standard form. The above polynomials h_{11}, h_{12}, h_{13} are chosen such that the conditions of Procedure 6.3.1 are satisfied. In particular, we want that the resulting polynomial p_1 should not leak information about the polynomials g_1, g_2, and g_4 used for computing p_1 and that it should look like a random non-commuting polynomial of A_3 with a sufficient high degree as compared to $d_g = \max\{\deg(g) \mid g \in G_T\}$.

For instance, since $\deg(g_1) = 14$ and $\mathrm{LT}_\sigma(g_1) = x_1^4x_3^{10}$, for h_{11}, we choose a random term $t = x_1^9x_3^5\partial_1^5\partial_3^3$ of degree 22. Now the leading term of the product $t\,g_1$ is $x_1^{13}x_3^{15}\partial_1^5\partial_3^3$ and to cancel it so that it does not appear in p_1, we set another term $t' = x_1^{13}x_3\partial_1^5\partial_3^3$ for h_{12}. If required, we proceed the same way for cancelling the terms in $\mathrm{DF}(t\,g_1)$. Note that now we have $\mathrm{DF}(t\,g_2) = x_1^{13}x_3^{15}\partial_1^5\partial_3^3$ and it will not appear in p_1. Continuing this way, we keep on adding and setting various terms for h_{11}, h_{12}, and h_{13} and finally compute p_1 as above. In this way, many terms in p_1 are either cancelled or their coefficients are changed. This can be easily seen by observing the number of terms in the homogeneous components of $h_{11}\,g_1$, $h_{12}\,g_2$, and $h_{13}\,g_4$ and comparing them with the number of terms of the homogeneous components of p_1, for instance, by using a CAS.

Similarly, choose

$$
\begin{aligned}
h_{21} &= x_1^3x_2^3x_3^4\partial_1^3\partial_2^3\partial_3 + x_1^3x_2^2x_3^4\partial_1^3\partial_2^2\partial_3 + x_1^2x_2^3x_3^4\partial_1^2\partial_2^3\partial_3 + x_1^4x_2^4x_3^4\partial_3 + \\
&\quad x_1^2x_2^2x_3^4\partial_1^2\partial_2^2\partial_3 + x_1^6x_3^4\partial_1\partial_2 + x_1^6x_3^4 + x_1^6x_2^3\partial_3 + x_1^4x_2^2\partial_3 + x_3^4, \\
h_{22} &= x_1^3x_2^5\partial_1^3\partial_2^3\partial_3 + x_1^3x_2^3x_3^2\partial_1^3\partial_2^3\partial_3 + x_1^4x_2^6\partial_3 + x_1^6x_2^2\partial_1\partial_2 + x_1^6x_2^2 + x_2^2, \\
h_{23} &= x_2^5x_3^6\partial_3 + x_3^2\partial_3 + \partial_1\partial_2 + 1, \\
h_{31} &= x_1x_2^3\partial_1 + x_2^4\partial_1 + x_1x_2x_3^2\partial_1 + x_1x_2^2\partial_1\partial_2 + x_1x_2^3 + x_2^2\partial_1 + \partial_1, \\
h_{32} &= x_1x_2x_3\partial_1^3\partial_2^3\partial_3^9 + x_1x_2\partial_1^3\partial_2^3\partial_3^8 + x_1x_3\partial_1^3\partial_2^2\partial_3^9 + x_2x_3\partial_1^2\partial_2^3\partial_3^9 + \\
&\quad x_1\partial_1^3\partial_2^2\partial_3^8 + x_2\partial_1^2\partial_2^3\partial_3^8 + x_3\partial_1^2\partial_2^2\partial_3^9 + \partial_1^2\partial_2^2\partial_3^8 + x_2^6\partial_1^2\partial_2^2 + \partial_1\partial_3^9 + \\
&\quad \partial_3^9 + x_2^6\partial_1\partial_2 + x_2^6, \\
h_{33} &= \partial_1^6\partial_2^4\partial_3 + x_1x_2x_3\partial_1^3\partial_2^3\partial_3 + \partial_1^7\partial_2\partial_3 + x_1x_2\partial_1^3\partial_2^3 + x_1x_3\partial_1^3\partial_2^2\partial_3 + \\
&\quad x_2x_3\partial_1^2\partial_2^3\partial_3 + x_1\partial_1^3\partial_2^2 + x_2\partial_1^2\partial_2^3 + x_3\partial_1^2\partial_2^2\partial_3 + \partial_1^2\partial_2^2 + \partial_1^3\partial_3 + \partial_3, \\
h_{34} &= \partial_1^8\partial_2^6 + \partial_1^7\partial_2^5 + x_1^5x_2\partial_1 + x_1^4x_2^2\partial_1 + x_1^5\partial_1\partial_2 + x_1^5x_2 + x_1^4\partial_1 + 1,
\end{aligned}
$$

and then compute

$$
\begin{aligned}
p_2 &= h_{21}\,g_3 + h_{22}\,g_6 + h_{23}\,g_7, \\
p_3 &= h_{31}\,g_5 + h_{32}\,g_8 + h_{33}\,g_9 + h_{34}\,g_{10}.
\end{aligned}
$$

The polynomial p_2 has degree 27 and consists of 148 terms in its standard form. The polynomial p_3 has degree 28 and 126 terms in its standard form. The polynomials h_{ij} are chosen such that the highest degree forms during the computation of the polynomials p_i cancel. Moreover, the leading terms of the polynomials in G_T are difficult to guess from the polynomials $p_1, p_2,$ and p_3 of the public key Q. To increase the member of lower degree terms in p_2 and p_3, we can now use Step (4) of Procedure 6.4.1 as follows: Choose $q_2' = x_2 + 1, q_3' = x_1$ and replace p_2 and p_3 by $p_2 q_2'$ and $p_3 q_3'$. The number of terms, respectively, in the standard forms of the new replaced polynomials p_2 and p_3 is 290 and 166 respectively, and $\deg(p2) = 28$, $\deg(p_3) = 29$.

We set the public key as $Q = \{p_1, p_2, p_3\}$. These public polynomials are given in Appendix C.3.

(3) **The Message Space:**
For the message space we choose

$$\mathscr{M} = \{x^\alpha \partial^\beta \mid |\alpha| + |\beta| \leq 4\}$$

That is, $\langle \mathscr{M} \rangle_K$ is the vector space of all polynomials in A_3 of degree less than or equal to 4. With this $\mathscr{M}$, we can have 2^{210} possible plaintext messages. This message space is also known publicly.

This message space fulfils Condition (2) of Procedure 6.4.1, i.e. every polynomials in G_T has at least one element from $\mathscr{O}_\sigma(I_T) \setminus \mathscr{M}$.

(4) **Encryption:**
Suppose that the plaintext message $m \in \langle \mathscr{M} \rangle_K$ is given by the following polynomial

$$\begin{aligned} m \;=\;& x_1x_2x_3\partial_1 + x_1x_2\partial_1^2 + x_2x_3\partial_1^2 + x_1x_2\partial_1\partial_2 + x_2^2\partial_1\partial_2 + x_1x_3\partial_1\partial_2 + x_1\partial_1\partial_2^2 \\ & + x_2^3\partial_3 + x_3\partial_1^2\partial_3 + x_1x_3\partial_2\partial_3 + x_3\partial_2^2\partial_3 + \partial_1\partial_2\partial_3^2 + x_2x_3\partial_1 + x_3\partial_1^2 + x_1^2\partial_2 \\ & + x_2^2\partial_2 + x_2x_3\partial_3 + x_1\partial_1\partial_3 + x_2\partial_3^2 + x_1x_2 + x_2\partial_2 + \partial_1. \end{aligned}$$

For the encryption, choose

$$\begin{aligned} \ell_1 &= x_1^6x_2^2x_3^3\partial_1^9\partial_2^3\partial_3^9 + 1, \qquad r_1 = x_1^5x_2^3x_3^3\partial_1^5\partial_2^2\partial_3^5 + x_1x_2 + x_3, \\ \ell_2 &= x_1^{10}x_2x_3^4\partial_1^{11}\partial_2\partial_3^{11} + \partial_2\partial_3 + \partial_3 + 1, \\ r_2 &= x_1^9x_3^3\partial_1^5\partial_2\partial_3^5 + x_1^3x_2^3 + x_1\partial_1 + x_3 + 1, \\ \ell_3 &= \partial_1^3\partial_2^3\partial_3^3 + \partial_1\partial_3^5 + \partial_2\partial_3 + \partial_1, \qquad r_3 = x_1^3x_2^3x_3^3 + x_1^3x_2, \end{aligned}$$

$$\begin{aligned}
\ell_4 &= x_1^3x_2^3x_3^3\partial_1^3\partial_2^3\partial_3^3, & \ell_5 &= x_1x_2\partial_1 + x_2x_3\partial_1 + \partial_2,\\
\ell_6 &= x_1x_2x_3 + x_1x_2\partial_1 + x_2x_3\partial_1 + x_3\partial_2, & r_7 &= x_1^{11}x_2^4x_3^6\partial_1^{14}\partial_2^5\partial_3^{14},\\
r_8 &= x_3\partial_1^2\partial_3 + x_3\partial_2^2\partial_3 + x_3\partial_1\partial_3 + x_1x_2 + x_2\partial_3 + \partial_3, & &\\
r_9 &= x_2^2\partial_1\partial_2 + x_3\partial_1 + \partial_2^2 + x_1 + x_2 + x_3 + 1, & &\\
r_4 &= r_5 = r_6 = \ell_7 = \ell_8 = \ell_9 = 1. & &
\end{aligned}$$

and compute the ciphertext c as the standard form of

$$c = \ell_1p_1r_1 + \ell_2p_2r_2 + \ell_3p_3r_3 + \ell_4p_1 + \ell_5p_2 + \ell_6p_3 + p_1r_7 + p_2r_8 + p_3r_9 + m.$$

Note that by taking $r_4 = r_5 = r_6 = \ell_7 = \ell_8 = \ell_9 = 1$, we are using summands with only one-sided multiplication. The polynomial c then has degree 87 and there are 13,532 terms in its standard form. We have selected the polynomials $\ell_1,\dots,\ell_9$, and $r_1,\dots,r_9$ in the same way as described earlier in the encryption process of Example 6.3.2. In this way, the highest degree terms cancel and many other terms are either cancelled or their coefficients are changed in the middle and lower part of the resulting ciphertext. The lower degree parts of the ciphertext polynomial c are dense enough to include many terms from the set $\mathscr{M}$. In this way out of 22 monomials of m, 16 are cancelled or their coefficients are changed in the ciphertext c. The remaining 6 monomials of m are mixed among other 82 monomials of c from the message space.

(5) **Decryption:**
For recovering the plaintext message m we compute $\mathrm{NR}_{(\sigma,\mathscr{G}_T)}(c)$, the normal remainder of c modulo the Gröbner basis $\mathscr{G}_T$. An efficient implementation of the left Division Algorithm 2.3.18 can recover m within a few seconds. For instance, such an implementation on the CAS `Singular` takes 3.93 seconds on our computing machine for the decryption.

Observations: In the setting of such a TWGBC, the secret key G_T is contained in the center C_n. Since they are commuting polynomials of Weyl algebra, the creation of a key-pair is relatively easy as compared to WGBC. For instance, in the above example note the computation of the polynomials $p_1, p_2,$ and p_3. Here, *Bob*, only have to choose a polynomial h_{ij} as described in Procedure 6.3.1, such that no information about structure of the system of generators of the ideal I_T is visible unchanged. On the other hand, the sender *Alice* can mess-up the ciphertext by using suitably chosen Weyl polynomials both for the left and the right multiplication in the encryption process. It turns out that such a ciphertext can only be decrypted efficiently when the correct secret key, i.e. when a two-sided σ-Gröbner basis is at hand. As far as the attacker *Eve* is concerned, it seems that her only choice is to compute a complete two-sided Gröbner basis of the ideal $J = \langle p_1, p_2, p_3 \rangle_T \subset I_T$. But, on the basis of our experimental results, by using Algorithm 6.1.9 for computing a two-sided Gröbner basis, this task turns out to be infeasible for the attacker in the setting of TWGBC (see Section 6.4 for details).

Let us now create another concrete case of TWGBC with a Weyl algebra over a field of characteristic 3.

Example 6.4.4. Over the finite field $K = \mathbb{F}_3$, consider the Weyl algebra $A_3 = \mathbb{F}_3[x_1, x_2, x_3, \partial_1 . \partial_2, \partial_3]$ of index 3. Let the term ordering on B_3 be $\sigma = \mathtt{DegRevLex}$. Note that here the center is given by $C_3 = \mathbb{F}_3[x_1^3, x_2^3, x_3^3, \partial_1^3, \partial_2^3, \partial_3^3]$. With these ingredients, we introduce following TWGBC.

(1) **Secret Key:**
Choose the following polynomials of A_3

$$\begin{aligned} f_1 &= x_1^9 x_2^6 + x_1^6 x_2^3 + \partial_1^3 + 1, & f_2 &= x_2^9 + x_2^6 x_3^3 - x_2^3 + 1, \\ f_3 &= \partial_1^9 \partial_2^6 + \partial_1^6 \partial_2^3 + \partial_1^3 + 1, & f_4 &= \partial_3^{12} + x_1^3 \partial_2^3 \partial_3^3 + \partial_1^6 + 1, \\ f_5 &= x_3^{15} \partial_2^3 + \partial_3^9 - x_3^3 \partial_1^3 + \partial_2^3 + 1. \end{aligned}$$

Let $I_T = \langle f_1, f_2, f_3 f_4, f_5 \rangle$ be the two-sided ideal generated by these polynomials. Then the reduced two-sided σ-Gröbner basis G_T of I_T is the set $\{g_1, \ldots, g_{10}\}$ where

$$\begin{aligned} g_1 = \; & x_3^{30}\partial_1^3 + x_3^{30} - x_3^{18}\partial_1^9 + x_3^6\partial_1^{18}\partial_2^3 + \partial_1^{18}\partial_2^3\partial_3^6 - x_3^6\partial_1^{15}\partial_2^3 - \partial_1^{15}\partial_2^3\partial_3^6 - \\ & x_3^{15}\partial_3^9 + x_3^3\partial_1^{12}\partial_3^9 - x_1^3 x_3^{15}\partial_1^3 - x_3^{18}\partial_1^3 + x_3^{15}\partial_1^6 - x_3^6\partial_1^{15} - x_1^3 x_3^3\partial_1^{12}\partial_2^3 + \\ & x_3^3\partial_1^{15}\partial_2^3 + \partial_1^{12}\partial_2^3\partial_3^6 - x_1^3 x_3^{15} - x_3^{15}\partial_1^3 - x_3^6\partial_1^{12} - x_3^3\partial_1^{12}\partial_2^3 - \partial_1^{12}\partial_3^6 - \\ & \partial_1^9\partial_2^3\partial_3^6 - \partial_1^9\partial_3^9 - x_1^3 x_3^3\partial_1^9 - x_3^3\partial_1^{12} + x_1^3\partial_1^9\partial_2^3 + \partial_1^{12}\partial_2^3 + x_3^3\partial_1^9 - \partial_1^9\partial_2^3 \\ & -\partial_1^6\partial_3^6 + x_1^3\partial_3^9 + x_1^3\partial_1^6 - \partial_1^9 - \partial_3^9 - x_1^3\partial_1^3 - x_3^3\partial_1^3 - x_1^3 + \partial_1^3 - 1, \end{aligned}$$

$$
\begin{aligned}
g_2 &= x_3^{15}\partial_1^3\partial_3^3 + x_3^3\partial_1^{12}\partial_2^3\partial_3^3 + \partial_1^{15}\partial_2^3 + x_3^{15}\partial_3^3 + x_3^3\partial_1^9\partial_3^3 - \partial_1^9\partial_2^3\partial_3^3 + \partial_1^{12} + \\
&\quad \partial_1^9\partial_2^3 - x_1^3\partial_1^3\partial_3^3 - \partial_1^6\partial_3^3 + \partial_1^6 - x_1^3\partial_3^3 + \partial_1^3\partial_3^3 + \partial_3^3, \\
g_3 &= \partial_1^9\partial_2^3\partial_3^9 - x_3^{15}\partial_1^3 - x_3^3\partial_1^{12}\partial_2^3 - x_3^{15} + \partial_1^6\partial_3^9 - x_3^3\partial_1^9 + \partial_1^9\partial_2^3 + \partial_1^6 - \\
&\quad \partial_1^3 - 1, \\
g_4 &= x_1^{12} - x_1^3x_2^6\partial_1^3 - x_1^3x_2^3x_3^3\partial_1^3 - x_2^6x_3^3\partial_1^3 - x_2^3x_3^6\partial_1^3 + x_1^9 + x_1^6x_2^3 - x_1^3x_2^6 + \\
&\quad x_1^6x_3^3 - x_1^3x_2^3x_3^3 - x_2^6x_3^3 - x_2^3x_3^6 - x_1^3x_2^3\partial_1^3 - x_1^3x_3^3\partial_1^3 - x_1^3x_2^3 - x_1^3x_3^3 + \\
&\quad x_1^3\partial_1^3 + x_3^3\partial_1^3 + x_1^3 + x_3^3 + \partial_1^3 + 1, \\
g_5 &= x_1^6x_2^6 + x_1^6x_2^3x_3^3 - x_1^9 + x_2^6\partial_1^3 + x_2^3x_3^3\partial_1^3 - x_1^6 + x_2^6 + x_2^3x_3^3 + x_2^3\partial_1^3 \\
&\quad + x_3^3\partial_1^3 + x_2^3 + x_3^3 - \partial_1^3 - 1, \\
g_6 &= x_3^{15}\partial_2^3 + \partial_3^9 - x_3^3\partial_1^3 + \partial_2^3 + 1, \\
g_7 &= x_1^9x_2^3 - x_2^6\partial_1^3 - x_2^3x_3^3\partial_1^3 + x_1^6 - x_2^6 - x_2^3x_3^3 + \partial_1^3 + 1, \\
g_8 &= \partial_1^9\partial_2^6 + \partial_1^6\partial_2^3 + \partial_1^3 + 1, \\
g_9 &= \partial_3^{12} + x_1^3\partial_2^3\partial_3^3 + \partial_1^6 + 1, \\
g_{10} &= x_2^9 + x_2^6x_3^3 - x_2^3 + 1.
\end{aligned}
$$

The secret key is the set G_T and the set $\mathscr{O}_\sigma(I_T)$ is also kept secret. Let $\mathscr{G}_T = (g_1, \ldots, g_{10})$.

(2) **Public Key:**

Let us now create polynomials p_1, p_2 for the public key Q by using some polynomials in G_T. As described in Example 6.4.3, choose

$$
\begin{aligned}
h_{11} &= x_3\partial_1^7\partial_2^7 + x_1\partial_1\partial_2\partial_3^5 - x_3\partial_3^5 + \partial_1^2\partial_2\partial_3^2, \\
h_{12} &= -x_1x_3^{15}\partial_1\partial_2\partial_3^2 - x_3^{15}\partial_1^2\partial_2\partial_3^2 + x_3^{16}\partial_3^2 + \partial_1^6\partial_2^9 + \partial_1^3\partial_2^6, \\
h_{13} &= -x_3^{31}\partial_1\partial_2 - x_3^{15}\partial_2^3\partial_3^3 - \partial_1^3\partial_2^3 + 1, \\
h_{21} &= -x_1\partial_1^7\partial_2^9 + \partial_1^3\partial_2^3 + x_1\partial_1\partial_2^3 + \partial_1\partial_2\partial_3^3 - \partial_3^5 + \partial_2^3\partial_3 + 1, \\
h_{22} &= -x_1\partial_1^{10}\partial_2^6 + x_1\partial_1^4 - \partial_3^4, \quad h_{23} = -\partial_1^{10}\partial_2^4 + \partial_1^9\partial_2^3\partial_3^2 + \partial_3, \\
h_{24} &= x_1\partial_1^7\partial_2^6\partial_3^9 - \partial_1^3\partial_3^9 - \partial_3^{10},
\end{aligned}
$$

and then compute the standard form of

$$
\begin{aligned}
p_1 &= h_{11}\,g_1 + h_{12}\,g_2 + h_{13}\,g_8 \\
p_2 &= h_{21}\,g_3 + h_{22}\,g_6 + h_{23}\,g_7 + h_{24}\,g_8.
\end{aligned}
$$

The polynomial p_1 has degree 45 and consists of 203 terms in its standard form. The polynomial p_2 has degree 35 and there are 91 terms in its standard form. The polynomials h_{ij} are chosen (as the way described in Example 6.4.3) such that the highest degree forms of the polynomials p_i are cancelled. To make p_2 more random looking, we can use Step (4) of Procedure 6.4.1 as follows: choose $h'_2 = \partial_1, q'_3 = x_1x_3$ and replace p_2 by $h'_2 p_2 q'_2$. The polynomial p_2 has degree 38 and there are 258 terms in its standard form.

We set the public key $Q = \{p_1, p_2\}$. These public polynomials are given in Appendix C.3.

(3) **The Message Space:**
For the message space we choose

$$\mathscr{M} = \{x^\alpha \partial^\beta \mid |\alpha| + |\beta| \leq 8\}$$

That is, $\langle \mathscr{M} \rangle_K$ is the vector space of all polynomials in A_3 of degree less than or equal to 8. With this $\mathscr{M}$, we can have 3^{3003} possible plaintext messages. As usual, $\mathscr{M}$ is known publicly. Moreover, every polynomial in G_T has at least one element from $\mathscr{O}_\sigma(I_T) \setminus \mathscr{M}$.

(4) **Encryption:**
Suppose that the plaintext message $m \in \langle \mathscr{M} \rangle_K$ is given by the polynomial

$$\begin{aligned} m \;=\; & x_2^3x_3^2\partial_2^2 - x_2^2x_3^3\partial_1\partial_3 + x_2^3x_3\partial_1\partial_2\partial_3 - x_2x_3\partial_1\partial_2^3\partial_3 + x_1\partial_1^3\partial_2\partial_3^2 + x_2^2\partial_2^3\partial_3^2 \\ & - x_2^2\partial_2^3 + \partial_2^5 + x_1^2x_2\partial_2\partial_3 - x_1x_2x_3\partial_2 - x_1x_3\partial_2^2 + x_1^2x_2\partial_3 + x_1x_2x_3\partial_3 - \\ & x_1x_3\partial_3^2 + x_1\partial_1^2 + x_1x_2\partial_2 + x_1x_3\partial_2 + x_3^2\partial_2 + x_1\partial_1. \end{aligned}$$

For the encryption, choose

$$\begin{aligned} \ell_1 &= x_1^2x_2^3x_3^2\partial_1^2\partial_2^4, & r_1 &= x_1^2x_2^4x_3^2\partial_1^{16}\partial_2^4, \\ \ell_2 &= -x_1x_2^3x_3^{15}\partial_1^2\partial_2 + \partial_3^2, & r_2 &= x_1x_2^4x_3^{16}\partial_1^3\partial_2^2 + x_2^2, \end{aligned}$$
$$\begin{aligned} \ell_3 &= x_2^3x_3^2\partial_2^2 - x_2^2x_3^3\partial_1\partial_3 + x_2^3x_3\partial_1\partial_2\partial_3 - x_2x_3\partial_1\partial_2^3\partial_3 - x_1x_2\partial_2, \\ \ell_4 &= x_1\partial_2 + x_3\partial_2 + \partial_1\partial_2 - x_1 + \partial_1 - 1, \\ r_6 &= x_1x_3^4 - \partial_1^2\partial_2\partial_3^2 + x_2^2\partial_3^2 - x_2^2 + \partial_2^2, \\ r_3 &= r_4 = r_5 = \ell_5 = \ell_6 = 1. \end{aligned}$$

Next, we compute the ciphertext c as the standard form of

$$c = \ell_1 p_1 r_1 + \ell_2 p_2 r_2 + \ell_3 p_1 r_3 + \ell_4 p_2 r_4 + \ell_5 p_1 r_5 + \ell_6 p_2 r_6 + m$$

Then the polynomial c has degree 84 and there are 8,557 terms in its support. We have selected the polynomials $\ell_1, \ldots, \ell_6$, and $r_1, \ldots, r_6$ in the same way as described in the encryption process of Example 6.3.2. In this way, the highest degree terms cancel and many other terms are either cancelled or their coefficients are changed in the resulting ciphertext. The lower part of the ciphertext polynomial c is dense enough to include many terms from the set $\mathscr{M}$ and the monomials of the plaintext message m are either cancelled or their coefficients are changed in the ciphertext c. In this way out of 19 monomials of m, 13 are cancelled from the ciphertext c. The remaining 6 monomials of m are mixed in 282 monomials of the message space that are present in c. Therefore, m is well-hidden in c.

(5) **Decryption:**
For recovering the plaintext message m, we compute $\mathrm{NR}_{(\sigma,\mathscr{G}_T)}(c)$. An efficient implementation of the left Division Algorithm 2.3.18 can recover m within a second. For instance, such an implementation in the CAS `Singular` takes 0.63 seconds on our computing machine for the decryption.

In the next section, we shall discuss the security of these instances of TWGBC against known standard attacks.

6.5 Efficiency and Security

As explained in Chapter 5, efficient algorithms are available for the computation in Weyl algebras both for positive and zero characteristic. In particular, both *Alice* and *Bob* can compute effectively in the setting of TWGBC for the encryption and decryption processes respectively. For a TWGBC, the key-generation is rather faster than the key-generation process of WGBC, since, by construction, the polynomials in the secret key G_T are elements of the commutative polynomial ring $C_n = \mathbb{F}_p[x_1^p, \ldots, x_n^p, \partial_1^p, \ldots, \partial_n^p]$. Therefore, in this case, *Bob* can easily control the sizes of the supports of polynomials $p_1, \ldots, p_s$ in his public key. Note here that $p_1, \ldots, p_s \notin C_n$, and therefore the sender *Alice* has to perform several Weyl multiplications for the encryption. Recall that, for encrypting a plaintext message $m \in \langle \mathscr{M} \rangle_K$, *Alice* has to compute the ciphertext c as the standard form of

$$c = \sum_{i=1}^{s'} \ell_i \, p_{k_i} \, r_i, \;\text{ where } s' \geq s \text{ and } k_i \in \{1, ..., s\} \tag{*}$$

In the computation of c, both left and right Weyl multiplication of polynomials are involved. This is of course a plus point for a TWGBC. In this setting, the TWGBC environment seems to be more favourable for the users of the cryptosystem. The process of converting the resulting polynomials into their standard form after both the left and the right multiplication provides sufficient flexibility to hide the polynomials that are used for the encryption. Contrary to the general non-commutative setting of GBC, this is very interesting phenomenon of TWGBC and we, therefore, explain it further in the following remark.

Remark 6.5.1 (TWGBC and non-commutative Polly Cracker). Our proposed TWGBC has a major advantage over Rai's basic non-commutative Polly Cracker cryptosystem. In our setting of TWGBC, we are multiplying a polynomial p_i from the left side by a polynomial ℓ_i and from the right side by a polynomial r_i. Then we convert the product $\ell_i \, p_{k_i} \, r_i$ into its standard form, where, as before $k_i \in \{1, \ldots, s\}$. Therefore, for a term $t \in \mathrm{Supp}(p_{k_i})$, an attacker will have difficulties to guess which terms $t_\ell \in \mathrm{Supp}(\ell_i)$ and $t_r \in \mathrm{Supp}(r_i)$ was used for the left and the right multiplication by the term t. This will become more difficult to guess from the ciphertext polynomial c when various such summands are combined, as in the Equation (*) above.

This favourable environment of TWGBC might also reduce its efficiency by increasing the size of the support of c to a value that may results in a bad 'data-rate' for transmitting c over a network. Therefore, users of TWGBC have to be very careful in choosing various polynomials in Equation (*) for the encryption. Note that, the aim for the encryption is to hide the plaintext message m and also to make c random-looking, so that the polynomials used for the encryption become difficult to guess from the ciphertext. For controlling the size of $\mathrm{Supp}(c)$, we have suggested in Remark 6.3.3 to use a finite field $\mathbb{F}_p$ such that $p \leq 7$. Moreover, we also suggest in the above Equation (*) to use most of the summands with only one-sided multiplication with p_{k_i} by taking one of ℓ_i or r_i as 1. For the summands where the polynomials ℓ_j and r_j are used for the left as well as the right multiplication with p_{k_j}, keep the sizes of the supports of ℓ_j and r_j as low as possible. We illustrate this by the following example.

Example 6.5.2. Consider the instance of TWGBC of Example 6.4.3. Let the plaintext message m and the polynomial p_1, p_2, p_3 be given as in Example 6.4.3. For encrypting the message m, choose

$$
\begin{aligned}
\ell_1 &= x_1^6 x_2^2 x_3^3 \partial_1^9 \partial_2^3 \partial_3^9, & r_1 &= x_1^5 x_2^3 x_3^3 \partial_1^5 \partial_2^2 \partial_3^5,\\
\ell_2 &= x_1^{10} x_2 x_3^4 \partial_1^{11} \partial_2 \partial_3^{11}, & r_2 &= x_1^9 x_3^3 \partial_1^5 \partial_2 \partial_3^5,\\
\ell_3 &= \partial_1^3 \partial_2^3 \partial_3^3 + \partial_1 \partial_2 \partial_3, & r_3 &= x_1^2 x_2^3 x_3^3 + x_1^3 x_2 + x_2^3 x_3,\\
\ell_4 &= \partial_1 + \partial_2 + 1, & \ell_5 &= x_2 x_3 \partial_1 + \partial_1^2 + \partial_2 \partial_3 + \partial_3 + 1,\\
\ell_6 &= x_1 x_2 x_3 + x_1 x_2 \partial_1 + x_2 x_3 \partial_1, & &\\
r_7 &= x_3 \partial_1^2 \partial_3 + x_1 x_2 + x_1 x_3 + x_1 \partial_1 + x_2 \partial_3 + x_1 + x_2 + x_3, & &\\
r_8 &= x_3 \partial_1 \partial_2 \partial_3 + x_1 x_3 \partial_1 + x_3 \partial_1^2 + x_3 \partial_1 \partial_2 + x_3 \partial_1 \partial_3 + x_1 \partial_2 \partial_3 + x_2 \partial_2 \partial_3 + x_3 \partial_2 \partial_3 + & &\\
&\quad x_1^2 + x_1 x_2 + x_1 x_3 + x_1 \partial_1 + x_2 \partial_1 + x_3 \partial_1 + x_1 \partial_2 + x_2 \partial_2 + x_3 \partial_2 + \partial_1 \partial_2 + x_1 \partial_3 + & &\\
&\quad x_2 \partial_3 + x_3 \partial_3 + \partial_2 \partial_3 + x_1 + 1, & &\\
r_4 &= r_5 = r_6 = \ell_7 = \ell_8 = 1, & &
\end{aligned}
$$

and compute the ciphertext c as

$$c = m + \ell_1 p_1 r_1 + \ell_2 p_2 r_2 + \ell_3 p_3 r_3 + \ell_4 p_1 + \ell_5 p_2 + \ell_6 p_3 + p_2 r_7 + p_3 r_8.$$

Note that by taking $r_4 = r_5 = r_6 = \ell_7 = \ell_8 = 1$ in the above representation of c, we are using only one-sided multiplication in the last 5 summands. The polynomial c then has degree 88 and number of terms in its support is reduced to 8890 from 13,532 (see Example 6.4.3). Moreover, the lower part of the ciphertext polynomial c is dense enough to include many terms from the message space and that the message m is also well-hidden, i.e. again, out of 22 monomials of m, 15 are cancelled from the ciphertext c and other 7 monomials are mixed among 82 monomials in c that are from the message space. Simultaneously, the decryption time is reduced to 2.9 seconds on our computing machine.

Hence the efficiency issue arising from the growth of the ciphertext polynomial is somewhat controllable by using the above suggestions for encryption and by choosing a base field of small characteristic. Of course, it also depends on the size s, the number of polynomials in the public key and the sizes of the supports of these polynomials. The instances of TWGBC that have been presented in Examples 6.3.2, 6.4.3, and 6.4.4, have decryption time of 0.79, 3.93 and 0.63 seconds respectively on our computing machine. There is strong evidence that our proposed TWGBC is efficient in terms of the amount of time required to legally decrypt the ciphertext and to recover the plaintext message m. For these instances of TWGBC, we have achieved data-rates of $1/337,\ 1/615,$ and $1/450$ respectively. For the case of TWGBC shown in Example 6.4.3, we have seen in Example 6.5.2, that by changing the polynomials used for the encryption, the size of the resulting ciphertext can be controlled to improve the efficiency both in terms of decryption time and the data-rate. In this case, the data-rate is improved to approx. $1/400$ and the decryption time has been reduced to 2.9 second. To sum up, the efficiency of TWGBC, in terms of data-rate for transmitting the ciphertext seems to be reasonable as compared to the instances of usual CGBC that have been presented so far. We believe that further investigation might result in better ways to control the size of the resulting ciphertext and hence to improve the data-rate for transmission.

On the other hand, the set-up of TWGBC gives us more security and reliability as compared to WGBC. We have already seen in Chapter 5, that hard instances of WGBC can be formulated that seem to be secure against the known standard attacks. Let us now discuss the security of TWGBC against these attacks:

(1) **Linear Algebra Attacks:** For the WGBC case, we have described in Section 5.2 that hard instances of WGBC can be formulated that are secure against the attacks based on linear algebra. For instance, in this setting, we have seen that for the instances of WGBC presented in Chapter 4, these attacks are not practical to apply, because the resulting linear system of equations turns out to be hard to solve. In contrast, there is no room for such attacks on TWGBC (see Remark 6.2.4), i.e. an instance of TWGBC is not vulnerable to Attacks 5.2.1 and 5.2.4.

(2) **The Chosen Ciphertext Attack:** As in the case of WGBC, the basic setup of TWGBC provides security against Attack 5.4, since every polynomial $g \in G_T$ is chosen such that $\mathrm{Supp}(g)$ contains at least one term from $\mathscr{O}_\sigma(I_T) \setminus \mathscr{M}$, where I_T is the two-sided ideal on which the instance of TWGBC is based. Hence Step (7) of Procedure 6.3.1 ensures that the basic chosen ciphertext attack will not be successful for a TWGBC, because of its built-in mechanism of recognizing an 'illegal' or 'fake' ciphertext (see Section 5.4 for details on how this attack works).

(3) **Partial Gröbner Basis Attack and TWGBC:** This attack on an instance of TWGBC works exactly the same way as described in Section 5.3 in the setting of WGBC. In the setting of TWGBC, the computation of a two-sided partial Gröbner basis, even for the degree bound that is less than the required by the attack, turns out to be more harder than for the cases of WGBC. Our experimental results give a strong evidence that a partial Gröbner basis attack is infeasible to apply on an instance of TWGBC based on Procedure 6.4.1 (see the examples below).

We now give computational evidence that the partial Gröbner basis attack is infeasible for the instances of TWGBC presented in Examples 6.3.2, 6.4.3 and 6.4.4.

Example 6.5.3. For the instance of TWGBC of Example 6.3.2, let $J = \langle p_1, p_2 \rangle_T$ be the two-sided ideal generated by the Weyl polynomials p_1 and p_2 of the public key Q. In this case, we have $\deg(c) = 93$, where c is the ciphertext polynomial. Let us now attempt to attack this system by computing a partial two-sided Gröbner basis of J. For this, we first try to compute a left partial Gröbner basis of the ideal J using the CAS `Singular` for the degree bound 85. This computation takes more than 56 hours of CPU time on our 'computing machine, consumes 4.4 GB of memory, and returns a partial left Gröbner basis consisting of 817 polynomials.

On the other hand, for the same value of the degree bound, a two-sided partial Gröbner basis is found to be infeasible. In fact, we terminated the computation after 10512.4 minutes of CPU time and utilizing more than 7 GB of memory. Since c is computed in a two-sided ideal, its normal remainder with respect to a complete or a partial left Gröbner basis cannot be equal to the plaintext message m. In the present case, computation of the normal remainder resulted in a polynomial of degree 84 and its standard form contains 120535 terms. The time taken by this computation was 12.1 hours on our computing machine. On the basis of these observations, we conclude that the partial Gröbner basis attack does not work on this instance of TWGBC.

Example 6.5.4. Consider the TWGBC presented in Example 6.4.3. In this case, for the ciphertext polynomial c we have $\deg(c) = 88$. Let $J = \langle p_1, p_2, p_3 \rangle_T$ be the two sided ideal of $A_3 = \mathbb{F}_2[x_1,x_2,x_3,\partial_1,\partial_2,\partial_3]$ generated by the polynomials in the public key $Q = \{p_1, p_2, p_3\}$ of this system. Again, the partial Gröbner basis attack on this system does not work, since a partial Gröbner basis for the degree bound 50 is found to be hard to compute. Note that for the possibility of success of this attack, an attacker has to compute a partial Gröbner basis for a degree bound larger than 50. In the present case, for the degree bound 50, the memory consumed during the computation on the CAS `Singular` grows to 4.1 GB in 643.24 minutes of CPU time on our computing machine. Hence there is sufficient evidence that, for a value larger than the degree bound, the computation of a partial two-sided Gröbner basis is infeasible.

Example 6.5.5. For the TWGBC of Example 6.4.4, a partial Gröbner basis attack fails as follows: The computation of a two-sided partial Gröbner basis of the ideal $J = \langle p_1, p_2 \rangle_T$ is found to be infeasible, where $p_1, p_2 \in A_3 = \mathbb{F}_3[x_1,x_2,x_3,\partial_1,\partial_2,\partial_3]$ are as given in Example 6.4.4. In this case, for the degree bound 71, our computation had grown to consume 2.2 GB of memory in 74.4 minutes and remained busy in the reduction process for the next 1220 minutes. We terminated our computations without an output after 1294.43 minutes of CPU time on our computing machine.

The computational results and observations obtained from the above examples are sufficient to conclude that there is strong evidence that a partial Gröbner basis attack can be ignored safely for the instances of TWGBC that are based on Procedure 6.4.1.

Conclusion: To conclude this thesis, we believe that hard instances of our proposed WGBC and TWGBC can be constructed such that they will have resistance against known standard attacks proposed by cryptanalyst of Gröbner basis type cryptosystems. The underlying problem of these systems is the computation of Gröbner basis of ideals of Weyl algebras. that is known to be EXPSPACE hard in general (see [53]). Therefore, Gröbner basis type cryptosystems do not have a threat of 'quantum computing' like RSA and ElGamal cryptosystems.

The cryptanalysis of such cryptosystems might be helpful in exploring the structure of their base rings, i.e. Weyl algebras. For instance, one might come up with new ideas and the modification of known attacks or some interesting algorithmic results for computations in Weyl algebras. In particular, a faster and more efficient way to compute a two-sided Gröbner basis of two-sided ideals of Weyl algebras will be a good contribution. Our examples presented in this chapter can be used to check the timings, efficiency and complexity of these new algorithms. Further investigation of these cryptosystems might also result in suggesting better ways of controlling the size of the ciphertext c and improving the efficiency of these systems, but not at the cost of security. A positive solution could be to minimize the sizes of the supports of polynomials $p_1, \ldots, p_s$ in public key such that computation of a left (resp. two-sided) Gröbner basis of the left (resp. two-sided) ideal J generated by these polynomial remains infeasible. Currently, to the best of our knowledge of the subject, we believe that these systems are reliable and might be adapted for the secret communication. We support our claim by all our experimental results, observations and examples presented in this thesis and by the challenges presented in the next section.

6.6 TWGBC Challenge:

Challenge 6.6.1. Over the field $K = \mathbb{F}_3$, consider the Weyl algebra $A_3 = \mathbb{F}_3[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$. Let the term ordering $\sigma = \texttt{DegRevLex}$ on the set of terms B_3 of A_3. We introduce the following TWGBC

(1) **Secret Key**
The secret key is the reduced two-sided σ-Gröbner basis G of a two-sided ideal $I_T \subset A_3$.

(2) **Public Key**
The set $Q = \{p_1, p_2, p_3\}$ is our public key, where

$$
\begin{aligned}
p_1 = {} & x_1^5x_2^5x_3^2\partial_1^{10}\partial_2^4\partial_3^{14} - x_2x_3^{10}\partial_1^9\partial_2^{19} - x_2^7x_3^{10}\partial_2^{22} - x_2^4x_3^4\partial_1^9\partial_2^{22} - x_2^{10}x_3^4\partial_2^{25} + x_1^3x_2^4x_3^{16}\partial_2^{13}\partial_3^3 - \\
& x_1^3x_2^{10}x_3^4\partial_2^{19}\partial_3^3 + x_1^3x_2^4x_3^{10}\partial_2^{10}\partial_3^{12} - x_2^4x_3^{10}\partial_1^3\partial_2^{10}\partial_3^{12} + x_1^5x_2^5x_3^2\partial_1^9\partial_2^4\partial_3^{14} - x_1^{11}x_2^4x_3^5\partial_1^2\partial_2^{14}\partial_3 - x_1^5x_2x_3^5\partial_1^{11}\partial_2^{14}\partial_3 - \\
& x_1^5x_2^7x_3^5\partial_1^2\partial_2^{17}\partial_3 - x_1^{14}x_2^4x_3^5\partial_1^2\partial_2^8\partial_3^4 - x_1^8x_2x_3^5\partial_1^{11}\partial_2^8\partial_3^4 - x_1^8x_2^7x_3^5\partial_1^2\partial_2^{11}\partial_3^4 - x_1^8x_2^4x_3^5\partial_1^2\partial_2^5\partial_3^{13} + x_1^5x_2^4x_3^5\partial_1^5\partial_2^5\partial_3^{13} + \\
& x_2^4x_3^{16}\partial_1^6\partial_2^{10} - x_2^4x_3^4\partial_1^{15}\partial_2^{13} - x_2^{10}x_3^4\partial_1^6\partial_2^{16} - x_2^7x_3\partial_1^9\partial_2^{19} - x_2^{13}x_3\partial_2^{22} + x_1^{11}x_2^4x_3^5\partial_1\partial_2^{14}\partial_3 + x_1^5x_2x_3^5\partial_1^{10}\partial_2^{14}\partial_3 + \\
& x_1^5x_2^7x_3^5\partial_1\partial_2^{17}\partial_3 + x_1^3x_2^{10}x_3^7\partial_2^{13}\partial_3^3 - x_1^3x_2^{13}x_3\partial_2^{16}\partial_3^3 + x_1^{14}x_2^4x_3^5\partial_1\partial_2^8\partial_3^4 + x_1^8x_2x_3^5\partial_1^{10}\partial_2^8\partial_3^4 + x_1^8x_2^7x_3^5\partial_1\partial_2^{11}\partial_3^4 + \\
& x_1^3x_2^4x_3^{10}\partial_1^3\partial_2^4\partial_3^{12} + x_2^3x_3^6\partial_1^9\partial_2^6\partial_3^{12} + x_2^9x_3^6\partial_2^6\partial_3^{12} - x_2^4x_3^{10}\partial_2^{10}\partial_3^{12} + x_1^3x_2^4x_3^4\partial_2^{13}\partial_3^{12} + x_1^8x_2^4x_3^5\partial_1\partial_2^5\partial_3^{13} - \\
& x_1^5x_2^4x_3^5\partial_1^4\partial_2^5\partial_3^{13} - x_1^{11}x_2^4x_3^5\partial_1^8\partial_2^5\partial_3 - x_1^5x_2x_3^5\partial_1^{17}\partial_2^5\partial_3 - x_1^5x_2^7x_3^5\partial_1^8\partial_2^8\partial_3 - x_1^{11}x_2^5x_3^2\partial_1\partial_2^{13}\partial_3^2 - x_1^5x_2^8x_3^2\partial_1\partial_2^{16}\partial_3^2 - \\
& x_1^{14}x_2^5x_3^2\partial_1\partial_2^7\partial_3^5 - x_1^8x_2^8x_3^2\partial_1\partial_2^{10}\partial_3^5 + x_1^5x_2^4x_3^5\partial_1^2\partial_2^5\partial_3^{13} - x_1^8x_2^5x_3^2\partial_1\partial_2^4\partial_3^{14} + x_1^5x_2^5x_3^2\partial_1^4\partial_2^4\partial_3^{14} + x_2^{10}x_3^7\partial_1^6\partial_2^{10} - \\
& x_2^7x_3\partial_1^{15}\partial_2^{10} - x_2^{13}x_3\partial_1^6\partial_2^{13} + x_2^3x_3^{15}\partial_2^{15} - x_2^3x_3^3\partial_1^9\partial_2^{18} + x_1^3x_2x_3^{10}\partial_2^{19} - x_2x_3^{10}\partial_1^3\partial_2^{19} + x_2^4x_3\partial_1^9\partial_2^{19} - \\
& x_2^9x_3^3\partial_2^{21} - x_1^3x_2^7x_3\partial_2^{22} + x_2^{10}x_3\partial_2^{22} + x_1^3x_2^4x_3^4\partial_2^{22} - x_2^4x_3^4\partial_1^3\partial_2^{22} + x_1^{11}x_2^4x_3^5\partial_1^7\partial_2^5\partial_3 + x_1^5x_2x_3^5\partial_1^{16}\partial_2^5\partial_3 + \\
& x_1^5x_2^7x_3^5\partial_1^7\partial_2^8\partial_3 - x_1^{11}x_2^5x_3^2\partial_2^{13}\partial_3^2 - x_1^5x_2^8x_3^2\partial_2^{16}\partial_3^2 + x_1^3x_2^3x_3^{15}\partial_2^9\partial_3^3 - x_1^3x_2^7x_3^7\partial_2^{13}\partial_3^3 + x_2^4x_3^{13}\partial_2^{13}\partial_3^3 - \\
& x_1^3x_2^9x_3^3\partial_2^{15}\partial_3^3 + x_1^3x_2^{10}x_3\partial_2^{16}\partial_3^3 + x_2^4x_3^{10}\partial_2^{16}\partial_3^3 - x_1^3x_2^4x_3^4\partial_1^3\partial_2^{16}\partial_3^3 + x_2^4x_3\partial_1^9\partial_2^{16}\partial_3^3 + x_2^{10}x_3\partial_2^{19}\partial_3^3 - \\
& x_1^{14}x_2^5x_3^2\partial_2^7\partial_3^5 - x_1^8x_2^8x_3^2\partial_2^{10}\partial_3^5 + x_2^{12}x_3^3\partial_2^6\partial_3^{12} + x_1^3x_2^3x_3^9\partial_2^6\partial_3^{12} - x_2^3x_3^9\partial_1^3\partial_2^6\partial_3^{12} - x_1^5x_2^4x_3^5\partial_1\partial_2^5\partial_3^{13} - \\
& x_1^8x_2^5x_3^2\partial_2^4\partial_3^{14} + x_1^5x_2^5x_3^2\partial_1^3\partial_2^4\partial_3^{14} - x_1^{11}x_2^5x_3^2\partial_1^7\partial_2^4\partial_3^2 - x_1^5x_2^8x_3^2\partial_1^7\partial_2^7\partial_3^2 + x_1^5x_2^5x_3^2\partial_1\partial_2^4\partial_3^{14} + x_2^3x_3^{15}\partial_1^6\partial_2^6 - \\
& x_2^3x_3^3\partial_1^{15}\partial_2^9 - x_2^4x_3^{16}\partial_2^{10} + x_1^3x_2^4x_3^7\partial_1^6\partial_2^{10} - x_2^7x_3^7\partial_1^6\partial_2^{10} + x_2^4x_3\partial_1^{15}\partial_2^{10} - x_2^9x_3^3\partial_1^6\partial_2^{12} - x_1^3x_2^7x_3\partial_1^6\partial_2^{13} + \\
& x_2^{10}x_3\partial_1^6\partial_2^{13} + x_1^3x_2^4x_3^4\partial_1^6\partial_2^{13} + x_2^9x_3^6\partial_2^{15} - x_2^3x_3^{12}\partial_2^{15} - x_2^6\partial_1^9\partial_2^{15} + x_2^{10}x_3^4\partial_2^{16} - x_2^{12}\partial_2^{18} + x_1^6x_2^4x_3\partial_2^{19} - \\
& x_2x_3^{10}\partial_2^{19} - x_2^4x_3^4\partial_2^{22} - x_1^{11}x_2^5x_3^2\partial_1^6\partial_2^4\partial_3^2 - x_1^5x_2^8x_3^2\partial_1^6\partial_2^7\partial_3^2 + x_1^3x_2^9x_3^6\partial_2^9\partial_3^3 - x_1^3x_2^3x_3^{12}\partial_2^9\partial_3^3 + x_2^4x_3^{10}\partial_1^3\partial_2^{10}\partial_3^3 - \\
& x_1^3x_2^{12}\partial_2^{12}\partial_3^3 + x_1^9x_2^4x_3\partial_2^{13}\partial_3^3 + x_2^{10}x_3^4\partial_2^{13}\partial_3^3 + x_1^3x_2^4x_3^7\partial_2^{13}\partial_3^3 + x_2^{10}x_3\partial_2^{16}\partial_3^3 - x_1^3x_2^4x_3^4\partial_2^{16}\partial_3^3 + x_2^4x_3^4\partial_2^{19}\partial_3^3 + \\
& x_1^3x_2^3x_3^9\partial_1^3\partial_3^{12} - x_1^3x_2^4x_3^4\partial_1^3\partial_2^4\partial_3^{12} + x_1^3x_2^6x_3^3\partial_2^6\partial_3^{12} - x_2^9x_3^3\partial_2^6\partial_3^{12} - x_1^3x_2^3x_3^6\partial_2^6\partial_3^{12} - x_2^3x_3^9\partial_2^6\partial_3^{12} + x_2^3x_3^6\partial_1^3\partial_2^6\partial_3^{12} + \\
& x_1^3x_2^3x_3^3\partial_2^9\partial_3^{12} + x_1^3x_2^4x_3\partial_2^{10}\partial_3^{12} - x_2^4x_3\partial_1^3\partial_2^{10}\partial_3^{12} - x_2^4x_3\partial_2^{13}\partial_3^{12} + x_1^5x_2^5x_3^2\partial_2^4\partial_3^{14} + x_1^{11}x_2^4x_3^5\partial_1^2\partial_2^5\partial_3 + \\
& x_1^5x_2x_3^5\partial_1^{11}\partial_2^5\partial_3 + x_1^5x_2^7x_3^5\partial_1^2\partial_2^8\partial_3 + x_2^9x_3^6\partial_1^6\partial_2^6 - x_2^3x_3^{12}\partial_1^6\partial_2^6 - x_2^6\partial_1^{15}\partial_2^6 - x_2^{12}\partial_1^6\partial_2^9 - x_2^{10}x_3^7\partial_2^{10} + \\
& x_1^6x_2^4x_3\partial_1^6\partial_2^{10} + x_2^4x_3^7\partial_1^6\partial_2^{10} + x_2^7x_3\partial_1^9\partial_2^{10} + x_2^{13}x_3\partial_2^{13} + x_1^3x_2^7x_3^4\partial_2^{13} - x_1^3x_2^7x_3\partial_1^3\partial_2^{13} - x_2^4x_3^4\partial_1^6\partial_2^{13} + \\
& x_1^3x_2^3x_3^6\partial_2^{15} - x_2^6x_3^6\partial_2^{15} + x_2^3\partial_1^9\partial_2^{15} - x_1^3x_2^6\partial_2^{18} + x_2^9\partial_2^{18} + x_1^3x_2^3x_3^3\partial_2^{18} - x_2^3x_3^3\partial_1^3\partial_2^{18} - x_1^{11}x_2^4x_3^5\partial_1\partial_2^5\partial_3 - \\
& x_1^5x_2x_3^5\partial_1^{10}\partial_2^5\partial_3 - x_1^5x_2^7x_3^5\partial_1\partial_2^8\partial_3 - x_2^4x_3^4\partial_1^9\partial_2^7\partial_3^3 - x_1^3x_2^6x_3^6\partial_2^9\partial_3^3 + x_2^3x_3^{12}\partial_2^9\partial_3^3 + x_2^{10}x_3^4\partial_2^{10}\partial_3^3 + x_2^{10}x_3\partial_1^3\partial_2^{10}\partial_3^3 + \\
& x_1^3x_2^9\partial_2^{12}\partial_3^3 + x_2^3x_3^9\partial_2^{12}\partial_3^3 - x_1^3x_2^3x_3^3\partial_1^3\partial_2^{12}\partial_3^3 + x_2^3\partial_1^9\partial_2^{12}\partial_3^3 + x_1^3x_2^4x_3^4\partial_2^{13}\partial_3^3 - x_2^7x_3^4\partial_2^{13}\partial_3^3 + x_2^9\partial_2^{15}\partial_3^3 + \\
& x_2^4x_3\partial_1^3\partial_2^{16}\partial_3^3 + x_1^3x_2^4x_3^4\partial_2^4\partial_3^{12} + x_1^3x_2^4x_3\partial_1^3\partial_2^4\partial_3^{12} + x_2^6x_3^3\partial_2^6\partial_3^{12} + x_2^3x_3^6\partial_2^6\partial_3^{12} - x_2^4x_3\partial_2^{10}\partial_3^{12} + x_2^3x_3^9\partial_1^2\partial_3^{12} + \\
& x_1^2x_3^3\partial_1^9\partial_3^{12} - x_3^3\partial_1^{11}\partial_3^{12} + x_1^2x_2^6x_3^3\partial_2^3\partial_3^{12} - x_2^6x_3^3\partial_1^2\partial_2^3\partial_3^{12} + x_1^{11}x_2^5x_3^2\partial_1\partial_2^4\partial_3^2 + x_1^5x_2^8x_3^2\partial_1\partial_2^7\partial_3^2 - x_2^3x_3^9\partial_1\partial_3^{12} + \\
& x_3^3\partial_1^{10}\partial_3^{12} + x_2^6x_3^3\partial_1\partial_2^3\partial_3^{12} + x_1^3x_2^7x_3^4\partial_1^6\partial_2^4 - x_1^3x_2^7x_3\partial_1^9\partial_2^4 - x_2^3x_3^{15}\partial_2^6 + x_1^3x_2^3x_3^6\partial_1^6\partial_2^6 - x_2^6x_3^6\partial_1^6\partial_2^6 + \\
& x_2^3\partial_1^{15}\partial_2^6 - x_1^3x_2^6\partial_1^6\partial_2^9 + x_2^9\partial_1^6\partial_2^9 + x_1^3x_2^3x_3^3\partial_1^6\partial_2^9 - x_1^3x_2^4x_3^7\partial_2^{10} + x_2^7x_3^7\partial_2^{10} - x_2^4x_3\partial_1^9\partial_2^{10} + x_2^9x_3^3\partial_2^{12} + \\
& x_1^3x_2^7x_3\partial_2^{13} - x_2^{10}x_3\partial_2^{13} - x_1^3x_2^4x_3^4\partial_2^{13} + x_1^3x_2^4x_3\partial_1^3\partial_2^{13} + x_2^4x_3^4\partial_1^3\partial_2^{13} + x_1^6x_2^3\partial_2^{15} + x_2^3x_3^6\partial_2^{15} - x_2^3x_3^3\partial_2^{18} + \\
& x_1^{11}x_2^5x_3^2\partial_2^4\partial_3^2 + x_1^5x_2^8x_3^2\partial_2^7\partial_3^2 + x_2^3x_3^9\partial_1^3\partial_2^6\partial_3^3 - x_2^{13}x_3\partial_2^7\partial_3^3 + x_1^9x_2^3\partial_2^9\partial_3^3 + x_2^9x_3^3\partial_2^9\partial_3^3 + x_1^3x_2^3x_3^6\partial_2^9\partial_3^3 + \\
& x_1^3x_2^4x_3\partial_1^3\partial_2^{10}\partial_3^3 - x_2^7x_3\partial_1^3\partial_2^{10}\partial_3^3 - x_2^4x_3^4\partial_1^3\partial_2^{10}\partial_3^3 + x_2^9\partial_2^{12}\partial_3^3 - x_1^3x_2^3x_3^3\partial_2^{12}\partial_3^3 + x_2^4x_3^4\partial_2^{13}\partial_3^3 + x_2^3x_3^3\partial_2^{15}\partial_3^3 +
\end{aligned}
$$

$$x_2^4x_3\partial_2^{16}\partial_3^3 - x_1^3x_2^3x_3^3\partial_1^3\partial_3^{12} + x_2^4x_3\partial_1^3\partial_2^4\partial_3^{12} + x_1^3x_2^3\partial_2^6\partial_3^{12} - x_2^3\partial_1^3\partial_2^6\partial_3^{12} - x_2^3\partial_2^9\partial_3^{12} + x_1^2x_2^9\partial_3^{12} -$$
$$x_2^9\partial_1^2\partial_3^{12} + x_2^9\partial_1\partial_3^{12} + x_1^3x_2^4x_3\partial_1^9\partial_2^4 - x_2^9x_3^6\partial_2^6 + x_2^3x_3^{12}\partial_2^6 + x_1^6x_2^3\partial_1^6\partial_2^6 + x_2^3x_3^6\partial_1^6\partial_2^6 + x_2^6\partial_1^9\partial_2^6 +$$
$$x_2^{12}\partial_2^9 + x_1^3x_2^6x_3^3\partial_2^9 - x_1^3x_2^6\partial_1^3\partial_2^9 - x_2^3x_3^3\partial_1^6\partial_2^9 - x_1^6x_2^4x_3\partial_2^{10} - x_2^4x_3^7\partial_2^{10} - x_2^7x_3\partial_2^{13} + x_2^4x_3^4\partial_2^{13} - x_2^3x_3^3\partial_1^9\partial_2^3\partial_3^3 +$$
$$x_2^9x_3^3\partial_2^6\partial_3^3 + x_2^9\partial_1^3\partial_2^6\partial_3^3 - x_1^3x_2^7x_3\partial_2^7\partial_3^3 + x_2^{10}x_3\partial_2^7\partial_3^3 + x_1^3x_2^4x_3^4\partial_2^7\partial_3^3 - x_2^7x_3\partial_1^3\partial_2^7\partial_3^3 - x_2^4x_3^4\partial_1^3\partial_2^7\partial_3^3 +$$
$$x_1^3x_2^3x_3^3\partial_2^9\partial_3^3 - x_2^6x_3^3\partial_2^9\partial_3^3 + x_2^4x_3^4\partial_2^{10}\partial_3^3 + x_2^4x_3\partial_1^3\partial_2^{10}\partial_3^3 + x_2^3\partial_1^3\partial_2^{12}\partial_3^3 + x_1^3x_2^3x_3^3\partial_3^{12} + x_1^3x_2^3\partial_1^3\partial_3^{12} -$$
$$x_2^4x_3\partial_2^4\partial_3^{12} - x_2^3\partial_2^6\partial_3^{12} - x_1^2x_3^9\partial_2^9 - x_3^9\partial_1^2\partial_2^9 - x_1^5x_3^9\partial_2^3\partial_3^3 - x_1^3x_3^9\partial_1^2\partial_2^3\partial_3^3 - x_1^2x_2^6\partial_3^{12} - x_1^5x_3^3\partial_3^{12} -$$
$$x_1^3x_2^3\partial_1^2\partial_3^{12} + x_2^6\partial_1^2\partial_3^{12} + x_1^3x_3^3\partial_1^2\partial_3^{12} + x_1^2x_3^3\partial_1^3\partial_3^{12} - x_3^3\partial_1^5\partial_3^{12} + x_3^9\partial_1\partial_2^9 + x_1^3x_3^9\partial_1\partial_2^3\partial_3^3 + x_1^3x_2^3\partial_1\partial_3^{12} -$$
$$x_2^6\partial_1\partial_3^{12} - x_1^3x_3^3\partial_1\partial_3^{12} + x_3^3\partial_1^4\partial_3^{12} + x_1^3x_2^6x_3^3\partial_1^6 - x_1^3x_2^6\partial_1^9 - x_1^3x_2^7x_3^4\partial_2^4 + x_1^3x_2^7x_3\partial_1^3\partial_2^4 - x_2^7x_3\partial_1^6\partial_2^4 -$$
$$x_1^3x_2^3x_3^6\partial_2^6 + x_2^6x_3^6\partial_2^6 - x_2^3\partial_1^9\partial_2^6 + x_1^3x_2^6\partial_2^9 - x_2^9\partial_2^9 - x_1^3x_2^3x_3^3\partial_2^9 + x_1^3x_2^3\partial_1^3\partial_2^9 + x_2^3x_3^3\partial_1^3\partial_2^9 - x_2^{12}\partial_2^3\partial_3^3 +$$
$$x_1^3x_2^3\partial_1^3\partial_2^6\partial_3^3 - x_2^6\partial_1^3\partial_2^6\partial_3^3 - x_2^3x_3^3\partial_1^3\partial_2^6\partial_3^3 + x_2^7x_3\partial_2^7\partial_3^3 - x_2^4x_3^4\partial_2^7\partial_3^3 + x_2^4x_3\partial_1^3\partial_2^7\partial_3^3 + x_2^3x_3^3\partial_2^9\partial_3^3 +$$
$$x_2^3\partial_2^{12}\partial_3^3 + x_2^3\partial_1^3\partial_3^{12} - x_1^2x_3^9\partial_1^6 - x_3^9\partial_1^8 + x_1^2x_2^3\partial_3^{12} + x_1^2x_3^3\partial_3^{12} - x_3^3\partial_1^2\partial_3^{12} + x_3^9\partial_1^7 + x_3^3\partial_1\partial_3^{12} + x_1^3x_2^3\partial_1^9 -$$
$$x_1^3x_2^4x_3\partial_1^3\partial_2^4 - x_1^6x_2^3\partial_2^6 - x_2^3x_3^6\partial_2^6 - x_2^6\partial_2^9 + x_2^3x_3^3\partial_2^9 - x_1^3x_2^6\partial_2^3\partial_3^3 + x_2^9\partial_2^3\partial_3^3 + x_1^3x_2^3x_3^3\partial_2^3\partial_3^3 - x_2^6\partial_1^3\partial_2^3\partial_3^3 -$$
$$x_2^3x_3^3\partial_1^3\partial_2^3\partial_3^3 + x_2^3x_3^3\partial_2^6\partial_3^3 + x_2^3\partial_1^3\partial_2^6\partial_3^3 - x_2^4x_3\partial_2^7\partial_3^3 - x_1^5\partial_2^9 - x_1^8\partial_2^3\partial_3^3 - x_1^3x_2^6x_3^3 + x_1^3x_2^6\partial_1^3 - x_2^6\partial_1^6 +$$
$$x_2^7x_3\partial_2^4 + x_2^6\partial_2^3\partial_3^3 - x_2^3x_3^3\partial_2^3\partial_3^3 + x_2^3\partial_1^3\partial_2^3\partial_3^3 + x_1^2x_3^9 + x_3^9\partial_1^2 - x_1^5\partial_1^6 + \partial_1^2\partial_2^9 + x_1^3\partial_1^2\partial_2^3\partial_3^3 - x_3^9\partial_1 -$$
$$\partial_1\partial_2^9 - x_1^3\partial_1\partial_2^3\partial_3^3 - x_1^3x_2^3\partial_1^3 + \partial_2^9 + x_1^3\partial_2^3\partial_3^3 - x_2^3\partial_2^3\partial_3^3 + \partial_1^8 - \partial_1^7 + x_2^6 + \partial_1^6 + x_1^5 - \partial_1^2 + \partial_1 - 1,$$

$$p_2 = -x_1^5x_2^{11}x_3^2\partial_1^{15}\partial_2^{11}\partial_3 + x_1^5x_2^{10}x_3^2\partial_1^{15}\partial_2^{10}\partial_3 + x_1^4x_2^{11}x_3^2\partial_1^{14}\partial_2^{11}\partial_3 + x_1^5x_2^8x_3^2\partial_1^{15}\partial_2^{11}\partial_3 - x_1^4x_2^{10}x_3^2\partial_1^{14}\partial_2^{10}\partial_3 -$$
$$x_1^5x_2^7x_3^2\partial_1^{15}\partial_2^{10}\partial_3 - x_1^4x_2^8x_3^2\partial_1^{14}\partial_2^{11}\partial_3 - x_1^5x_2^5x_3^2\partial_1^{15}\partial_2^{11}\partial_3 - x_1^8x_2^5x_3^5\partial_1^6\partial_2^{14}\partial_3 + x_1^2x_2^5x_3^{11}\partial_1^6\partial_2^{14}\partial_3 +$$
$$x_1^4x_2^4\partial_1^{11}\partial_2^{19} + x_1^4x_2^7x_3^2\partial_1^{14}\partial_2^{10}\partial_3 + x_1^{13}x_2\partial_1^{11}\partial_2^{10}\partial_3^3 - x_1x_2\partial_1^{20}\partial_2^{13}\partial_3^3 - x_1x_2^7\partial_1^{11}\partial_2^{16}\partial_3^3 + x_1^7x_2\partial_1^{11}\partial_2^7\partial_3^{12} -$$
$$x_1^4x_2\partial_1^{14}\partial_2^7\partial_3^{12} + x_1^5x_2^4x_3^2\partial_1^{15}\partial_2^{10}\partial_3 + x_1^4x_2^5x_3^2\partial_1^{14}\partial_2^{11}\partial_3 + x_1^8x_2^4x_3^5\partial_1^6\partial_2^{13}\partial_3 - x_1^2x_2^4x_3^{11}\partial_1^6\partial_2^{13}\partial_3 + x_1^7x_2^5x_3^5\partial_1^5\partial_2^{14}\partial_3 -$$
$$x_1x_2^5x_3^{11}\partial_1^5\partial_2^{14}\partial_3 + x_1^4x_2^3\partial_1^{11}\partial_2^{18} - x_1^3x_2^4\partial_1^{11}\partial_2^{18} + x_1^3x_2^4\partial_1^{10}\partial_2^{19} + x_1^8x_2^{11}x_3^5\partial_1^6\partial_2^5\partial_3 - x_1^2x_2^{11}x_3^{11}\partial_1^6\partial_2^5\partial_3 -$$
$$x_1^8x_2^{11}x_3^2\partial_1^9\partial_2^5\partial_3 + x_1^{13}\partial_1^{11}\partial_2^9\partial_3^3 - x_1^{12}x_2\partial_1^{11}\partial_2^9\partial_3^3 + x_1^{12}x_2\partial_1^{10}\partial_2^{10}\partial_3^3 - x_1\partial_1^{20}\partial_2^{12}\partial_3^3 + x_2\partial_1^{20}\partial_2^{12}\partial_3^3 -$$
$$x_2\partial_1^{19}\partial_2^{13}\partial_3^3 - x_1x_2^6\partial_1^{11}\partial_2^{15}\partial_3^3 + x_2^7\partial_1^{11}\partial_2^{15}\partial_3^3 - x_2^7\partial_1^{10}\partial_2^{16}\partial_3^3 + x_1^7\partial_1^{11}\partial_2^6\partial_3^{12} - x_1^6x_2\partial_1^{11}\partial_2^6\partial_3^{12} -$$
$$x_1^4\partial_1^{14}\partial_2^6\partial_3^{12} + x_1^3x_2\partial_1^{14}\partial_2^6\partial_3^{12} + x_1^6x_2\partial_1^{10}\partial_2^7\partial_3^{12} - x_1^3x_2\partial_1^{13}\partial_2^7\partial_3^{12} + x_1^{10}x_2\partial_1^{17}\partial_2^7 + x_1^4x_2^4\partial_1^{17}\partial_2^{10} -$$
$$x_1^4x_2^4x_3^2\partial_1^{14}\partial_2^{10}\partial_3 - x_1^7x_2^4x_3^5\partial_1^5\partial_2^{13}\partial_3 + x_1x_2^4x_3^{11}\partial_1^5\partial_2^{13}\partial_3 + x_1^7x_2\partial_1^{14}\partial_2\partial_3^{12} - x_1^4x_2\partial_1^{11}\partial_2^7\partial_3^{12} + x_1x_2\partial_1^{11}\partial_2^{10}\partial_3^{12} +$$
$$x_1^3x_2^3\partial_1^{10}\partial_2^{18} - x_1^8x_2^{10}x_3^5\partial_1^6\partial_2^4\partial_3 + x_1^2x_2^{10}x_3^{11}\partial_1^6\partial_2^4\partial_3 + x_1^8x_2^{10}x_3^2\partial_1^9\partial_2^4\partial_3 - x_1^7x_2^{11}x_3^5\partial_1^5\partial_2^5\partial_3 + x_1x_2^{11}x_3^{11}\partial_1^5\partial_2^5\partial_3 +$$
$$x_1^7x_2^{11}x_3^2\partial_1^8\partial_2^5\partial_3 + x_1^{12}\partial_1^{10}\partial_2^9\partial_3^3 - \partial_1^{19}\partial_2^{12}\partial_3^3 - x_2^6\partial_1^{10}\partial_2^{15}\partial_3^3 + x_1^6\partial_1^{10}\partial_2^6\partial_3^{12} - x_1^3\partial_1^{13}\partial_2^6\partial_3^{12} + x_1^{10}\partial_1^{17}\partial_2^6 -$$
$$x_1^9x_2\partial_1^{17}\partial_2^6 + x_1^9x_2\partial_1^{16}\partial_2^7 + x_1^4x_2^3\partial_1^{17}\partial_2^9 - x_1^3x_2^4\partial_1^{17}\partial_2^9 + x_1^3x_2^4\partial_1^{16}\partial_2^{10} + x_1^8x_2^8x_3^2\partial_1^9\partial_2^5\partial_3 + x_1^8x_2^5x_3^5\partial_1^9\partial_2^5\partial_3 -$$
$$x_1^2x_2^5x_3^{11}\partial_1^9\partial_2^5\partial_3 + x_1^5x_2^5x_3^2\partial_1^6\partial_2^{14}\partial_3 + x_1^7\partial_1^{14}\partial_3^{12} - x_1^6x_2\partial_1^{14}\partial_3^{12} + x_1^6x_2\partial_1^{13}\partial_2\partial_3^{12} - x_1^4\partial_1^{11}\partial_2^6\partial_3^{12} +$$
$$x_1^3x_2\partial_1^{11}\partial_2^6\partial_3^{12} - x_1^3x_2\partial_1^{10}\partial_2^7\partial_3^{12} + x_1\partial_1^{11}\partial_2^9\partial_3^{12} - x_2\partial_1^{11}\partial_2^9\partial_3^{12} + x_2\partial_1^{10}\partial_2^{10}\partial_3^{12} + x_1^7x_2^{10}x_3^5\partial_1^5\partial_2^4\partial_3 -$$
$$x_1x_2^{10}x_3^{11}\partial_1^5\partial_2^4\partial_3 - x_1^7x_2^{10}x_3^2\partial_1^8\partial_2^4\partial_3 + x_1^4x_2\partial_1^{11}\partial_2^{13}\partial_3^3 - x_1x_2\partial_1^{14}\partial_2^{13}\partial_3^3 + x_1^9\partial_1^{16}\partial_2^6 + x_1^3x_2^3\partial_1^{16}\partial_2^9 -$$
$$x_1^8x_2^7x_3^2\partial_1^9\partial_2^4\partial_3 - x_1^8x_2^4x_3^5\partial_1^9\partial_2^4\partial_3 + x_1^2x_2^4x_3^{11}\partial_1^9\partial_2^4\partial_3 - x_1^7x_2^8x_3^2\partial_1^8\partial_2^5\partial_3 - x_1^7x_2^5x_3^5\partial_1^8\partial_2^5\partial_3 + x_1x_2^5x_3^{11}\partial_1^8\partial_2^5\partial_3 -$$

$$x_1^5x_2^4x_3^2\partial_1^6\partial_2^{13}\partial_3 - x_1^4x_2^5x_3^2\partial_1^5\partial_2^{14}\partial_3 + x_1^6\partial_1^{13}\partial_3^{12} - x_1^3\partial_1^{10}\partial_2^6\partial_3^{12} + \partial_1^{10}\partial_2^9\partial_3^{12} - x_1^4x_2x_3^9\partial_1^{10}\partial_2^6 - x_1^5x_2^{11}x_3^2\partial_1^6\partial_2^5\partial_3 -$$
$$x_1^8x_2^5x_3^5\partial_1^6\partial_2^5\partial_3 + x_1^2x_2^5x_3^{11}\partial_1^6\partial_2^5\partial_3 - x_1^8x_2^5x_3^2\partial_1^9\partial_2^5\partial_3 + x_1^4\partial_1^{11}\partial_2^{12}\partial_3^3 - x_1^3x_2\partial_1^{11}\partial_2^{12}\partial_3^3 - x_1\partial_1^{14}\partial_2^{12}\partial_3^3 +$$
$$x_2\partial_1^{14}\partial_2^{12}\partial_3^3 + x_1^3x_2\partial_1^{10}\partial_2^{13}\partial_3^3 - x_2\partial_1^{13}\partial_2^{13}\partial_3^3 - x_1^{10}x_2\partial_1^{11}\partial_2^7 - x_1^{13}x_2\partial_1^5\partial_2^{10} - x_1^4x_2^4\partial_1^{11}\partial_2^{10} - x_1^7x_2\partial_1^2\partial_2^{19} +$$
$$x_1^7x_2^7x_3^2\partial_1^8\partial_2^4\partial_3 + x_1^7x_2^4x_3^5\partial_1^8\partial_2^4\partial_3 - x_1x_2^4x_3^{11}\partial_1^8\partial_2^4\partial_3 + x_1^4x_2^4x_3^2\partial_1^5\partial_2^{13}\partial_3 - x_1x_2\partial_1^{11}\partial_2^{13}\partial_3^3 - x_1x_2\partial_1^{14}\partial_2\partial_3^{12} -$$
$$x_1^3x_2x_3^9\partial_1^9\partial_2^6 + x_1^4x_2^7x_3\partial_1^{10}\partial_2^6 + x_1^5x_2^{10}x_3^2\partial_1^6\partial_2^4\partial_3 + x_1^8x_2^4x_3^5\partial_1^6\partial_2^4\partial_3 - x_1^2x_2^4x_3^{11}\partial_1^6\partial_2^4\partial_3 + x_1^8x_2^4x_3^2\partial_1^9\partial_2^4\partial_3 +$$
$$x_1^4x_2^{11}x_3^2\partial_1^5\partial_2^5\partial_3 + x_1^7x_2^5x_3^5\partial_1^5\partial_2^5\partial_3 - x_1x_2^5x_3^{11}\partial_1^5\partial_2^5\partial_3 + x_1^7x_2^5x_3^2\partial_1^8\partial_2^5\partial_3 + x_1^3\partial_1^{10}\partial_2^{12}\partial_3^3 - \partial_1^{13}\partial_2^{12}\partial_3^3 -$$
$$x_1^4x_2^7\partial_1^{10}\partial_2^6 - x_1^{10}\partial_1^{11}\partial_2^6 + x_1^9x_2\partial_1^{11}\partial_2^6 + x_1^3x_2^7\partial_1^{11}\partial_2^6 - x_1^9x_2\partial_1^{10}\partial_2^7 - x_1^3x_2^7\partial_1^{10}\partial_2^7 - x_1^{13}\partial_1^5\partial_2^9 + x_1^{12}x_2\partial_1^5\partial_2^9 -$$
$$x_1^4x_2^3\partial_1^{11}\partial_2^9 + x_1^3x_2^4\partial_1^{11}\partial_2^9 - x_1^{12}x_2\partial_1^4\partial_2^{10} - x_1^3x_2^4\partial_1^{10}\partial_2^{10} - x_1^7\partial_1^2\partial_2^{18} + x_1^6x_2\partial_1^2\partial_2^{18} - x_1^6x_2\partial_1\partial_2^{19} -$$
$$x_1^5x_2^5x_3^2\partial_1^9\partial_2^5\partial_3 - x_1\partial_1^{11}\partial_2^{12}\partial_3^3 + x_2\partial_1^{11}\partial_2^{12}\partial_3^3 - x_2\partial_1^{10}\partial_2^{13}\partial_3^3 - x_1\partial_1^{14}\partial_3^{12} + x_2\partial_1^{14}\partial_3^{12} - x_2\partial_1^{13}\partial_2\partial_3^{12} +$$
$$x_1^3x_2^7x_3\partial_1^9\partial_2^6 + x_1^7x_2^7\partial_1^2\partial_2^{10} + x_1x_2^4\partial_1^{11}\partial_2^{10} - x_1^4x_2^{10}x_3^2\partial_1^5\partial_2^4\partial_3 - x_1^7x_2^4x_3^5\partial_1^5\partial_2^4\partial_3 + x_1x_2^4x_3^{11}\partial_1^5\partial_2^4\partial_3 -$$
$$x_1^7x_2^4x_3^2\partial_1^8\partial_2^4\partial_3 + x_1^4x_2^4\partial_1^{11}\partial_2^4\partial_3^3 + x_1x_2\partial_1^{11}\partial_2\partial_3^{12} - x_1^3x_2^7\partial_1^9\partial_2^6 - x_1^9\partial_1^{10}\partial_2^6 - x_1^3x_2^6\partial_1^{10}\partial_2^6 - x_1^4x_2^4x_3\partial_1^{10}\partial_2^6 -$$
$$x_1^{12}\partial_1^4\partial_2^9 - x_1^3x_2^3\partial_1^{10}\partial_2^9 - x_1^6\partial_1\partial_2^{18} + x_1^5x_2^4x_3^2\partial_1^9\partial_2^4\partial_3 + x_1^4x_2^5x_3^2\partial_1^8\partial_2^5\partial_3 - \partial_1^{10}\partial_2^{12}\partial_3^3 - \partial_1^{13}\partial_3^{12} + x_1^4x_2^4\partial_1^{10}\partial_2^6 -$$
$$x_1^3x_2^4\partial_1^{11}\partial_2^6 + x_1^3x_2^4\partial_1^{10}\partial_2^7 + x_1^7x_2^6\partial_1^2\partial_2^9 - x_1^6x_2^7\partial_1^2\partial_2^9 + x_1x_2^3\partial_1^{11}\partial_2^9 - x_2^4\partial_1^{11}\partial_2^9 + x_1^6x_2^7\partial_1\partial_2^{10} + x_2^4\partial_1^{10}\partial_2^{10} +$$
$$x_1^5x_2^5x_3^2\partial_1^6\partial_2^5\partial_3 + x_1^4x_2^3\partial_1^{11}\partial_2^3\partial_3^3 - x_1^3x_2^4\partial_1^{11}\partial_2^3\partial_3^3 + x_1^3x_2^4\partial_1^{10}\partial_2^4\partial_3^3 + x_1\partial_1^{11}\partial_3^{12} - x_2\partial_1^{11}\partial_3^{12} + x_2\partial_1^{10}\partial_2\partial_3^{12} +$$
$$x_1x_2^4\partial_1^{17}\partial_2 - x_1^3x_2^4x_3\partial_1^9\partial_2^6 + x_1^7x_2\partial_1^5\partial_2^{10} - x_1^4x_2^4x_3^2\partial_1^8\partial_2^4\partial_3 + x_1^3x_2^4\partial_1^9\partial_2^6 + x_1^3x_2^3\partial_1^{10}\partial_2^6 + x_1^4x_2x_3\partial_1^{10}\partial_2^6 +$$
$$x_1^6x_2^6\partial_1\partial_2^9 + x_1^7x_2x_3^4\partial_1\partial_2^9 - x_1x_2x_3^{10}\partial_1\partial_2^9 + x_2^3\partial_1^{10}\partial_2^9 - x_1^5x_2^4x_3^2\partial_1^6\partial_2^4\partial_3 - x_1^4x_2^5x_3^2\partial_1^5\partial_2^5\partial_3 + x_1^3x_2^3\partial_1^{10}\partial_2^3\partial_3^3 +$$
$$\partial_1^{10}\partial_3^{12} - x_1^7x_2x_3^9\partial_1^4 + x_1x_2^3\partial_1^{17} - x_2^4\partial_1^{17} + x_2^4\partial_1^{16}\partial_2 + x_1^3x_2\partial_1^{11}\partial_2^6 - x_1^3x_2\partial_1^{10}\partial_2^7 - x_1^7x_2x_3^3\partial_1\partial_2^9 +$$
$$x_1^6x_2x_3^3\partial_1^2\partial_2^9 - x_2x_3^9\partial_1^2\partial_2^9 + x_1^7\partial_1^5\partial_2^9 - x_1^6x_2\partial_1^5\partial_2^9 - x_1^6x_2x_3^3\partial_1\partial_2^{10} + x_2x_3^9\partial_1\partial_2^{10} + x_1^6x_2\partial_1^4\partial_2^{10} + x_1^3x_2x_3\partial_1^9\partial_2^6 +$$
$$x_1^6x_2x_3^4\partial_2^9 - x_2x_3^{10}\partial_2^9 - x_1^7x_2\partial_1^2\partial_2^{10} + x_1^4x_2^4x_3^2\partial_1^5\partial_2^4\partial_3 - x_1^7x_2^7x_3^4\partial_1 + x_1x_2^7x_3^{10}\partial_1 - x_1^6x_2x_3^9\partial_1^3 + x_1^7x_2^7x_3\partial_1^4 +$$
$$x_2^3\partial_1^{16} - x_1^3\partial_1^{10}\partial_2^6 - x_1^6x_2x_3^3\partial_2^9 - x_1^6x_3^3\partial_1\partial_2^9 + x_3^9\partial_1\partial_2^9 + x_1^6\partial_1^4\partial_2^9 + x_1^7x_2^7x_3^3\partial_1 - x_1^6x_2^7x_3^3\partial_1^2 + x_2^7x_3^9\partial_1^2 -$$
$$x_1^7x_2^7\partial_1^4 + x_1^6x_2^7\partial_1^5 + x_1^6x_2^7x_3^3\partial_1\partial_2 - x_2^7x_3^9\partial_1\partial_2 - x_1^6x_2^7\partial_1^4\partial_2 - x_1^7\partial_1^2\partial_2^9 + x_1^6x_2\partial_1^2\partial_2^9 - x_1^6x_2\partial_1\partial_2^{10} -$$
$$x_1^6x_2^7x_3^4 + x_2^7x_3^{10} + x_1^6x_2^7x_3\partial_1^3 - x_1x_2^4\partial_1^{11}\partial_2 + x_1^6x_2^7x_3^3 + x_1^6x_2^6x_3^3\partial_1 - x_2^6x_3^9\partial_1 - x_1^6x_2^7\partial_1^3 - x_1^6x_2^6\partial_1^4 -$$
$$x_1^7x_2^4x_3\partial_1^4 - x_1^7x_2x_3^4\partial_1^4 + x_1x_2x_3^{10}\partial_1^4 - x_1^6\partial_1\partial_2^9 - x_1^4x_2x_3\partial_1\partial_2^9 + x_1^7x_2^4\partial_1^4 + x_1^7x_2x_3^3\partial_1^4 - x_1^6x_2^4\partial_1^5 -$$
$$x_1^6x_2x_3^3\partial_1^5 + x_2x_3^9\partial_1^5 - x_1x_2^3\partial_1^{11} + x_2^4\partial_1^{11} + x_1^6x_2^4\partial_1^4\partial_2 + x_1^6x_2x_3^3\partial_1^4\partial_2 - x_2x_3^9\partial_1^4\partial_2 - x_2^4\partial_1^{10}\partial_2 + x_1^4x_2\partial_1\partial_2^9 -$$
$$x_1^3x_2\partial_1^2\partial_2^9 + x_1^3x_2\partial_1\partial_2^{10} - x_1^6x_2^4x_3\partial_1^3 - x_1^6x_2x_3^4\partial_1^3 + x_2x_3^{10}\partial_1^3 - x_1^3x_2x_3\partial_2^9 + x_1^4x_2^7x_3\partial_1 + x_1^7x_2x_3^4\partial_1 -$$
$$x_1x_2x_3^{10}\partial_1 + x_1^6x_2^4\partial_1^3 + x_1^6x_2x_3^3\partial_1^3 + x_1^6x_2^3\partial_1^4 + x_1^7x_2x_3\partial_1^4 + x_1^6x_3^3\partial_1^4 - x_3^9\partial_1^4 - x_2^3\partial_1^{10} + x_1^3x_2\partial_2^9 +$$
$$x_1^3\partial_1\partial_2^9 - x_1^4x_2^7\partial_1 - x_1^7x_2x_3^3\partial_1 + x_1^3x_2^7\partial_1^2 + x_1^6x_2x_3^3\partial_1^2 - x_2x_3^9\partial_1^2 + x_1^6x_2\partial_1^5 - x_1^3x_2^7\partial_1\partial_2 - x_1^6x_2x_3^3\partial_1\partial_2 +$$
$$x_2x_3^9\partial_1\partial_2 - x_1^6x_2\partial_1^4\partial_2 + x_1x_2\partial_1\partial_2^9 + x_1^3x_2^7x_3 + x_1^6x_2x_3^4 - x_2x_3^{10} + x_1^6x_2x_3\partial_1^3 - x_1^3x_2^7 - x_1^6x_2x_3^3 - x_1^3x_2^6\partial_1 -$$
$$x_1^6x_3^3\partial_1 + x_3^9\partial_1 - x_1^6\partial_1^4 + x_1^4x_2x_3\partial_1^4 + x_2\partial_2^9 - x_1x_2^7\partial_1 - x_1^4x_2\partial_1^4 + x_1^3x_2\partial_1^5 - x_1^3x_2\partial_1^4\partial_2 + x_1^3x_2x_3\partial_1^3 -$$
$$x_2^7 - x_1^4x_2x_3\partial_1 - x_1^3x_2\partial_1^3 - x_1^3\partial_1^4 + x_1^4x_2\partial_1 - x_1^3x_2\partial_1^2 - x_1x_2\partial_1^4 + x_1^3x_2\partial_1\partial_2 - x_1^3x_2x_3 + x_1^3x_2 +$$
$$x_1^3\partial_1 - x_2\partial_1^3 + x_1x_2\partial_1 + x_2,$$

$$
\begin{aligned}
p_3 = &-x_1^4x_2^{18}x_3^{11}\partial_1^2\partial_2^2\partial_3^2 - x_1^4x_2^{15}x_3^5\partial_1^{11}\partial_2^2\partial_3^2 - x_1^4x_2^{21}x_3^5\partial_1^2\partial_2^5\partial_3^2 - x_1^4x_2^{18}x_3^{11}\partial_1^2\partial_2^2\partial_3 - x_1^4x_2^{15}x_3^5\partial_1^{11}\partial_2^2\partial_3 - \\
&x_1^4x_2^{21}x_3^5\partial_1^2\partial_2^5\partial_3 - x_1^4x_2^{24}x_3^2\partial_1^2\partial_2^2\partial_3^2 - x_1^4x_2^{24}x_3^2\partial_1^2\partial_2^2\partial_3 - x_1^{12}x_2^3x_3^4\partial_1^9\partial_2^6 + x_1^6x_2^3x_3^{10}\partial_1^9\partial_2^6 + x_1^6x_2^3x_3^9\partial_1^9\partial_2^7 + \\
&x_1^6x_3^3\partial_1^{18}\partial_2^7 + x_1^6x_2^6x_3^3\partial_1^9\partial_2^{10} - x_1^{12}x_2^3x_3^3\partial_1^9\partial_2^6\partial_3 - x_1^6x_3^3\partial_1^{18}\partial_2^6\partial_3 - x_1^6x_2^6x_3^3\partial_1^9\partial_2^9\partial_3 + x_1x_3^{16}\partial_2^{16}\partial_3 + \\
&x_1^7x_2^3x_3^4\partial_2^{19}\partial_3 + x_1^4x_3^{16}\partial_2^{10}\partial_3^4 - x_1^4x_3^4\partial_1^9\partial_2^{13}\partial_3^4 - x_1^4x_2^6x_3^4\partial_2^{16}\partial_3^4 + x_1^4x_3^{10}\partial_2^7\partial_3^{13} - x_1^7x_3^4\partial_1^3\partial_2^7\partial_3^{13} + x_1^{12}x_2^3x_3^3\partial_1^9\partial_2^6 + \\
&x_1^6x_2^3x_3^9\partial_1^9\partial_2^6 - x_1^6x_3^3\partial_1^{18}\partial_2^6 - x_1^6x_2^6x_3^3\partial_1^9\partial_2^9 - x_1x_3^{16}\partial_2^{16} - x_1^7x_2^3x_3^4\partial_2^{19} - x_1^7x_2^{18}x_3^2\partial_1^2\partial_2^2\partial_3^2 + x_1^4x_2^{21}x_3^2\partial_1^2\partial_2^2\partial_3^2 + \\
&x_1^7x_2^{15}x_3^5\partial_1^2\partial_2^2\partial_3^2 - x_1^4x_2^{15}x_3^5\partial_1^5\partial_2^2\partial_3^2 - x_1^4x_3^{16}\partial_2^{10}\partial_3^3 + x_1^4x_3^4\partial_1^9\partial_2^{13}\partial_3^3 + x_1^4x_2^6x_3^4\partial_2^{16}\partial_3^3 - x_1^4x_3^{10}\partial_2^7\partial_3^{12} + \\
&x_1^7x_3^4\partial_1^3\partial_2^7\partial_3^{12} - x_1^7x_2^{18}x_3^2\partial_1^2\partial_2^2\partial_3 + x_1^4x_2^{21}x_3^2\partial_1^2\partial_2^2\partial_3 + x_1^7x_2^{15}x_3^5\partial_1^2\partial_2^2\partial_3 - x_1^4x_2^{15}x_3^5\partial_1^5\partial_2^2\partial_3 + x_1^6x_2^9x_3\partial_1^9\partial_2^6 + \\
&x_1^6x_2^9\partial_1^9\partial_2^7 + x_1^7x_3^{10}\partial_1^6\partial_2^7\partial_3 + x_1^7x_2^3x_3^4\partial_1^6\partial_2^{10}\partial_3 + x_1^7x_2^6x_3\partial_2^{16}\partial_3 + x_1x_2^6x_3^7\partial_2^{16}\partial_3 + x_1^2x_2^8x_3^{10}\partial_1^4\partial_2^5\partial_3^2 + \\
&x_1^2x_2^5x_3^4\partial_1^{13}\partial_2^5\partial_3^2 + x_1^2x_2^{11}x_3^4\partial_1^4\partial_2^8\partial_3^2 + x_1^4x_2^6x_3^7\partial_2^{10}\partial_3^4 - x_1^4x_2^3x_3\partial_1^9\partial_2^{10}\partial_3^4 - x_1^4x_2^9x_3\partial_2^{13}\partial_3^4 + x_1^4x_3^{10}\partial_1^3\partial_2\partial_3^{13} - \\
&x_1^7x_3^4\partial_2^7\partial_3^{13} + x_1^4x_3^4\partial_2^{10}\partial_3^{13} + x_1^6x_2^9\partial_1^9\partial_2^6 - x_1^7x_3^{10}\partial_1^6\partial_2^7 - x_1^7x_2^3x_3^4\partial_1^6\partial_2^{10} - x_1^7x_2^6x_3\partial_2^{16} - x_1x_2^6x_3^7\partial_2^{16} + \\
&x_1^2x_2^8x_3^{10}\partial_1^4\partial_2^5\partial_3 + x_1^2x_2^5x_3^4\partial_1^{13}\partial_2^5\partial_3 + x_1^2x_2^{11}x_3^4\partial_1^4\partial_2^8\partial_3 - x_1^4x_2^{18}x_3^2\partial_1^2\partial_2^2\partial_3^2 - x_1^4x_2^{15}x_3^5\partial_1^2\partial_2^2\partial_3^2 - x_1^4x_2^6x_3^7\partial_2^{10}\partial_3^3 + \\
&x_1^4x_2^3x_3\partial_1^9\partial_2^{10}\partial_3^3 + x_1^4x_2^9x_3\partial_2^{13}\partial_3^3 - x_1^4x_3^{10}\partial_1^3\partial_2\partial_3^{12} + x_1^7x_3^4\partial_2^7\partial_3^{12} - x_1^4x_3^4\partial_2^{10}\partial_3^{12} - x_1^4x_2^{18}x_3^2\partial_1^2\partial_2^2\partial_3 - \\
&x_1^4x_2^{15}x_3^5\partial_1^2\partial_2^2\partial_3 + x_1^9x_2^3x_3\partial_1^9\partial_2^6 - x_1^6x_2^6x_3\partial_1^9\partial_2^6 + x_1^9x_2^3\partial_1^9\partial_2^7 - x_1^6x_2^6\partial_1^9\partial_2^7 - x_1^9x_3^3\partial_1^9\partial_2^7 + x_1^6x_3^3\partial_1^{12}\partial_2^7 + \\
&x_1^9x_3^3\partial_1^9\partial_2^6\partial_3 - x_1^6x_3^3\partial_1^{12}\partial_2^6\partial_3 + x_1^7x_2^6x_3\partial_1^6\partial_2^7\partial_3 + x_1^{10}x_3\partial_2^{16}\partial_3 - x_1^7x_2^3x_3\partial_2^{16}\partial_3 + x_1^4x_3^7\partial_2^{16}\partial_3 - x_1x_2^3x_3^7\partial_2^{16}\partial_3 + \\
&x_1^2x_2^{14}x_3\partial_1^4\partial_2^5\partial_3^2 + x_1^{13}x_3\partial_2^{10}\partial_3^4 + x_1^7x_3^7\partial_2^{10}\partial_3^4 - x_1^4x_2^3x_3^7\partial_2^{10}\partial_3^4 + x_1^4x_3\partial_1^9\partial_2^{10}\partial_3^4 + x_1^4x_2^6x_3\partial_2^{13}\partial_3^4 + \\
&x_1^7x_3^4\partial_2^{13}\partial_3^4 - x_1^4x_3^4\partial_1^3\partial_2^{13}\partial_3^4 + x_1^7x_3\partial_2^7\partial_3^{13} + x_1^9x_2^3\partial_1^9\partial_2^6 - x_1^6x_2^6\partial_1^9\partial_2^6 + x_1^9x_3^3\partial_1^9\partial_2^6 - x_1^6x_3^3\partial_1^{12}\partial_2^6 - \\
&x_1^7x_2^6x_3\partial_1^6\partial_2^7 - x_1^{10}x_3\partial_2^{16} + x_1^7x_2^3x_3\partial_2^{16} - x_1^4x_3^7\partial_2^{16} + x_1x_2^3x_3^7\partial_2^{16} + x_1^2x_2^{14}x_3\partial_1^4\partial_2^5\partial_3 - x_1^{13}x_3\partial_2^{10}\partial_3^3 - \\
&x_1^7x_3^7\partial_2^{10}\partial_3^3 + x_1^4x_2^3x_3^7\partial_2^{10}\partial_3^3 - x_1^4x_3\partial_1^9\partial_2^{10}\partial_3^3 - x_1^4x_2^6x_3\partial_2^{13}\partial_3^3 - x_1^7x_3^4\partial_2^{13}\partial_3^3 + x_1^4x_3^4\partial_1^3\partial_2^{13}\partial_3^3 - x_1^7x_3\partial_2^7\partial_3^{12} + \\
&x_1^6x_3^3\partial_1^9\partial_2^7 + x_1^6x_2^3\partial_1^9\partial_2^6\partial_3 - x_1^6x_3^3\partial_1^9\partial_2^6\partial_3 - x_1^7x_3^{10}\partial_2^7\partial_3 + x_1^{10}x_3\partial_1^6\partial_2^7\partial_3 - x_1^7x_2^3x_3\partial_1^6\partial_2^7\partial_3 - x_1^7x_2^3x_3^4\partial_2^{10}\partial_3 + \\
&x_1^7x_3\partial_2^{16}\partial_3 - x_1^4x_2^3x_3\partial_2^{16}\partial_3 + x_1x_3^7\partial_2^{16}\partial_3 + x_1^5x_2^8x_3\partial_1^4\partial_2^5\partial_3^2 - x_1^2x_2^{11}x_3\partial_1^4\partial_2^5\partial_3^2 - x_1^5x_2^5x_3^4\partial_1^4\partial_2^5\partial_3^2 + \\
&x_1^2x_2^5x_3^4\partial_1^7\partial_2^5\partial_3^2 + x_1^{10}x_3\partial_2^{10}\partial_3^4 + x_1^4x_3^7\partial_2^{10}\partial_3^4 - x_1^4x_2^3x_3\partial_1^3\partial_2^{10}\partial_3^4 - x_1^4x_3^4\partial_2^{13}\partial_3^4 + x_1^7x_3\partial_1^3\partial_2\partial_3^{13} - x_1^4x_3^4\partial_1^3\partial_2\partial_3^{13} + \\
&x_1^4x_3\partial_2^7\partial_3^{13} - x_1^6x_2^3\partial_1^9\partial_2^6 - x_1^6x_3^3\partial_1^9\partial_2^6 + x_1^7x_3^{10}\partial_2^7 - x_1^{10}x_3\partial_1^6\partial_2^7 + x_1^7x_2^3x_3\partial_1^6\partial_2^7 + x_1^7x_2^3x_3^4\partial_2^{10} - x_1^7x_3\partial_2^{16} + \\
&x_1^4x_2^3x_3\partial_2^{16} - x_1x_3^7\partial_2^{16} + x_1^5x_2^8x_3\partial_1^4\partial_2^5\partial_3 - x_1^2x_2^{11}x_3\partial_1^4\partial_2^5\partial_3 - x_1^5x_2^5x_3^4\partial_1^4\partial_2^5\partial_3 + x_1^2x_2^5x_3^4\partial_1^7\partial_2^5\partial_3 - \\
&x_1^{10}x_3\partial_2^{10}\partial_3^3 - x_1^4x_3^7\partial_2^{10}\partial_3^3 + x_1^4x_2^3x_3\partial_1^3\partial_2^{10}\partial_3^3 + x_1^4x_3^4\partial_2^{13}\partial_3^3 - x_1^7x_3\partial_1^3\partial_2\partial_3^{12} + x_1^4x_3^4\partial_1^3\partial_2\partial_3^{12} - x_1^4x_3\partial_2^7\partial_3^{12} - \\
&x_1^7x_2^6x_3\partial_2^7\partial_3 + x_1^7x_3\partial_1^6\partial_2^7\partial_3 - x_1^4x_2^3x_3\partial_1^6\partial_2^7\partial_3 + x_1^4x_2^3x_3^4\partial_2^{10}\partial_3 - x_1^4x_2^3x_3\partial_1^3\partial_2^{10}\partial_3 + x_1^4x_3\partial_2^{16}\partial_3 + x_1^2x_2^8x_3\partial_1^4\partial_2^5\partial_3^2 + \\
&x_1^2x_2^5x_3^4\partial_1^4\partial_2^5\partial_3^2 + x_1^7x_2^3x_3^4\partial_2^4\partial_3^4 - x_1^7x_2^3x_3\partial_1^3\partial_2^4\partial_3^4 - x_1^4x_2^3x_3\partial_2^{10}\partial_3^4 + x_1^4x_3\partial_1^3\partial_2^{10}\partial_3^4 + x_1^4x_3^4\partial_2\partial_3^{13} + \\
&x_1^4x_3\partial_1^3\partial_2\partial_3^{13} + x_1^7x_2^6x_3\partial_2^7 - x_1^7x_3\partial_1^6\partial_2^7 + x_1^4x_2^3x_3\partial_1^6\partial_2^7 - x_1^4x_2^3x_3^4\partial_2^{10} + x_1^4x_2^3x_3\partial_1^3\partial_2^{10} - x_1^4x_3\partial_2^{16} + \\
&x_1^2x_2^8x_3\partial_1^4\partial_2^5\partial_3 + x_1^2x_2^5x_3^4\partial_1^4\partial_2^5\partial_3 - x_1^7x_2^3x_3^4\partial_2^4\partial_3^3 + x_1^7x_2^3x_3\partial_1^3\partial_2^4\partial_3^3 + x_1^4x_2^3x_3\partial_2^{10}\partial_3^3 - x_1^4x_3\partial_1^3\partial_2^{10}\partial_3^3 - \\
&x_1^4x_3^4\partial_2\partial_3^{12} - x_1^4x_3\partial_1^3\partial_2\partial_3^{12} + x_1^3x_3\partial_1^9\partial_2^6 + x_1^3\partial_1^9\partial_2^7 + x_1^4x_2^3x_3^4\partial_1^6\partial_2\partial_3 - x_1^4x_2^3x_3\partial_1^9\partial_2\partial_3 - x_1^3\partial_1^9\partial_2^6\partial_3 - \\
&x_1^{10}x_3\partial_2^7\partial_3 + x_1^7x_2^3x_3\partial_2^7\partial_3 + x_1^4x_3\partial_1^6\partial_2^7\partial_3 + x_1^4x_3\partial_1^3\partial_2^{10}\partial_3 + x_1^7x_3\partial_1^3\partial_2^4\partial_3^4 + x_1^4x_3\partial_2^{10}\partial_3^4 - x_1^4x_2^3x_3^4\partial_1^6\partial_2 + \\
&x_1^4x_2^3x_3\partial_1^9\partial_2 - x_1^3\partial_1^9\partial_2^6 + x_1^{10}x_3\partial_2^7 - x_1^7x_2^3x_3\partial_2^7 - x_1^4x_3\partial_1^6\partial_2^7 - x_1^4x_3\partial_1^3\partial_2^{10} - x_1^7x_3\partial_1^3\partial_2^4\partial_3^3 - x_1^4x_3\partial_2^{10}\partial_3^3 - \\
&x_1^3x_2^3x_3^9\partial_1\partial_2 - x_1^3x_3^3\partial_1^{10}\partial_2 - x_1^3x_2^6x_3^3\partial_1\partial_2^4 + x_1^3x_2^3x_3^9\partial_3^2 + x_1^3x_3^3\partial_1^9\partial_3^2 + x_1^3x_2^6x_3^3\partial_2^3\partial_3^2 + x_1^3x_3\partial_1^{12} + x_1^3\partial_1^{12}\partial_2 +
\end{aligned}
$$

$x_1^3x_2^6x_3\partial_1^3\partial_2^3+x_1^3x_2^6\partial_1^3\partial_2^4-x_1^3x_2^3x_3\partial_2^9-x_1^3x_2^3\partial_2^{10}+x_1^3x_2^3x_3^9\partial_3+x_1^3x_3^3\partial_1^9\partial_3-x_1^3\partial_1^{12}\partial_3+x_1^4x_3\partial_1^9\partial_2\partial_3+$
$x_1^3x_2^6x_3^3\partial_2^3\partial_3-x_1^3x_2^6\partial_1^3\partial_2^3\partial_3-x_1^7x_3\partial_2^7\partial_3+x_1^4x_2^3x_3\partial_2^7\partial_3+x_1^3x_2^3\partial_2^9\partial_3-x_1^3\partial_1^{12}-x_1^4x_3\partial_1^9\partial_2-x_1^3x_2^6\partial_1^3\partial_2^3+$
$x_1^7x_3\partial_2^7-x_1^4x_2^3x_3\partial_2^7+x_1^3x_2^3\partial_2^9-x_1^3x_2^9\partial_1\partial_2+x_1^3x_2^9\partial_3^2+x_1^3x_2^9x_3+x_1^3x_2^9\partial_2-x_1^4x_2^3x_3^4\partial_2\partial_3+x_1^4x_2^3x_3\partial_1^3\partial_2\partial_3-$
$x_1^4x_3\partial_2^7\partial_3-x_1^3x_2^9+x_1^4x_2^3x_3^4\partial_2-x_1^4x_2^3x_3\partial_1^3\partial_2+x_1^4x_3\partial_2^7-x_1^6x_2^3\partial_1\partial_2+x_1^3x_2^6\partial_1\partial_2+x_3^9\partial_1\partial_2-x_1^3x_3^3\partial_1^4\partial_2+$
$x_1^6x_2^3\partial_3^2-x_1^3x_2^6\partial_3^2-x_3^9\partial_3^2+x_1^3x_3^3\partial_1^3\partial_3^2+x_1^3x_2^3x_3\partial_1^3+x_1^3x_3\partial_1^6+x_1^3x_2^3\partial_1^3\partial_2+x_1^3\partial_1^6\partial_2+x_3\partial_2^9+\partial_2^{10}+$
$x_1^6x_2^3\partial_3-x_1^3x_2^6\partial_3-x_3^9\partial_3-x_1^3x_2^3\partial_1^3\partial_3+x_1^3x_3^3\partial_1^3\partial_3-x_1^3\partial_1^6\partial_3-x_1^4x_3\partial_1^3\partial_2\partial_3-\partial_2^9\partial_3-x_1^3x_2^3\partial_1^3-$
$x_1^3\partial_1^6+x_1^4x_3\partial_1^3\partial_2-\partial_2^9-x_1^3x_2^3\partial_1\partial_2+x_2^6\partial_1\partial_2-x_1^3x_3^3\partial_1\partial_2+x_1^3x_2^3\partial_3^2-x_2^6\partial_3^2+x_1^3x_3^3\partial_3^2-x_1^3x_2^3x_3-$
$x_2^6x_3+x_1^3x_3\partial_1^3-x_1^3x_2^3\partial_2-x_2^6\partial_2+x_1^3\partial_1^3\partial_2-x_1^3x_2^3\partial_3+x_1^3x_3^3\partial_3-x_1^3\partial_1^3\partial_3+x_1^3x_2^3+x_2^6-x_1^3\partial_1^3+$
$x_1^3\partial_1\partial_2-x_2^3\partial_1\partial_2-x_1^3\partial_3^2+x_2^3\partial_3^2-x_3\partial_1^3-\partial_1^3\partial_2-x_1^3\partial_3+x_2^3\partial_3+\partial_1^3\partial_3+\partial_1^3+\partial_1\partial_2-\partial_3^2+x_3+$
$\partial_2+\partial_3-1.$

(3) **Message Space** For the message space we choose

$$\mathscr{M}=\{x^\alpha\,\partial^\beta \;\mid\; |\alpha|+|\beta|\le 7\}$$

That is, $\langle\mathscr{M}\rangle_K$ is the vector space of all polynomials in A_3 of degree less than or equal to 7. With this $\mathscr{M}$, we can have 3^{1716} possible plaintext messages.

We have encrypted a message m and obtained the ciphertext c of degree 80 and its standard form consists of 9,703 terms. We believe that this ciphertext is secure and cannot be broken by using the known standard attacks presented in this thesis. The ciphertext c together with the public key Q is available in the file **twgbc_challenge.coc** in a format usable for the CAS ApCoCoA. This file can be downloaded from the WWW page

`http://www.megaupload.com/?d=54LD2L16`

We welcome our readers to attack this cryptosystem and provide us further useful suggestions and improvements. Keeping in mind the chosen-ciphertext security for the attack presented in Section 5.5, we are ready to decrypt any ciphertext message that is the encryption of a message in the following message space:

$$\mathscr{M}'=\{x^\alpha\,\partial^\beta \;\mid\; |\alpha|+|\beta|\le 4\}$$

Appendix A

Package Weyl

In Chapters 2 and 4 we talk about computations in Weyl algebra. In particular, we have defined the standard form of a Weyl polynomial and described the left Division Algorithm 2.3.18 for Weyl algebras. We have also explained algorithms for computing left and two-sided Gröbner bases of ideals in Weyl algebras (see Algorithms 2.3.24 and 6.1.9 for details). We have developed the package `Weyl` for performing various computations in Weyl algebras using ApCoCoA. In this appendix we are going to explain the usage of this package by briefly describing the functions which are implemented in this package for performing various computations in Weyl algebras. The CAS ApCoCoA, an acronym of 'Applied Computations in Commutative Algebra' is based on the CAS CoCoA. It is primarily designed for working with 'real-problems' by using the symbolic computations methods of CoCoAand by developing new libraries for related computations.

The CAS ApCoCoAis available free of charge via the internet and can be downloaded from the WWW page

`http://www.apcocoa.org/`

For a short introduction to CoCoA and for the help on getting started with it we refer to [27] (Appendix A, page 275). The ApCoCoAworks exactly the same way as explained there.

For working with the Weyl algebra of index n by using the CAS ApCoCoA, one first has to define and activate a ring in $2n$ indeterminates. For instance, for working with the Weyl algebra $A_5 = \mathbb{Z}_7[x_1,\dots,x_5,\partial_1,\dots,\partial_5]$ of index 5 one can start by using the following two commands:

```
An ::= ZZ/(7)[x[1..5],y[1..5]];
Use An;
```

Note that the symbol ∂ can be replaced by any other symbol that can be used to represent indeterminates in ApCoCoA. In general, given a ring in $2n$ indeterminates in ApCoCoA, the package `Weyl` takes the first n indeterminates as $x_1,\dots,x_n$ and the last n indeterminates as $\partial_1,\dots,\partial_n$ in the definition of the Weyl algebra A_n (see Definition 2.1.1). The default term ordering σ for the rings in ApCoCoA is defined as `DegRevLex`. For using other term orderings, see the ApCoCoA documentation from the help-menu.

A.1 Available Functions

In the following we give a short description of the functions available in the package `Weyl` for working with the Weyl Algebra A_n over a field K. This description is also available as part of the documentation of this package and can be seen from the help-menu of ApCoCoA.

A.1.1. `WStandardForm(L)`

Purpose: Computes the *standard form* of a Weyl polynomial.

Syntax `Weyl.WStandardForm(L:LIST):POLY`

Input A list `L` of lists where each list represents a monomial of a Weyl polynomial.

Output The standard form of the Weyl polynomial represented by the above list `L`.

Example Consider the Weyl algebra $A_2 = \mathbb{Q}[x_1,x_2,y_1,y_2]$. For converting a Weyl polynomial $F := 2x_2y_1x_2^2 - 9y_2x_1^2x_2^3 + 5$ in to its standard form, one has to run the following commands in ApCoCoA interactive window:

`A2::=QQ[x[1..2],y[1..2]];` -- Define the appropriate ring

`Use A2;`

`L := [ [2x[1],y[1],x[2]^2], [-9y[2],x[1]^2,x[2]^3],[5] ];`

– note how the polynomial F is represented by the above list `L`.

`Weyl.WStandardForm(L);`

```
-9x[1]^2x[2]^3y[2] - 27x[1]^2x[2]^2 + 2x[1]x[2]^2y[1] + 5
```

-- this output is the standard form of the given polynomial F.

Note. From now on, by a Weyl polynomial we mean a polynomial represented in its unique standard form. For using any of the function below, if a polynomial is not given in its standard form then first convert it into the standard form as explained above.

A.1.2. `WMulByMonom(M,P)`

Purpose: Computes the product `M*P` of a Weyl monomial `M` and a Weyl Polynomial `P`.

Syntax `Weyl.WMulByMonom(M:POLY,P:POLY):POLY`

Input 1st parameter `M`, a Weyl monomial in its standard form.
2nd parameter `P`, a Weyl polynomial.

Output The Weyl polynomial for the product `M*P`.

Example For multiplying a monomial $M = x^3y^4$ with the polynomial $F := x^3 + y^3 + 3xy + 5$, where both $M, F \in A_1 = \mathbb{Q}[x,y]$, We proceed as follows:

```
A1::=QQ[x,y]; Use A1; -- Define and activate the appropriate ring
M:=x^3y^4; F:=x^3+y^3+3xy+5;
Weyl.WMulByMonom(M,F);
x^6y^4+x^3y^7+3x^4y^5+12x^5y^3+17x^3y^4+36x^4y^2+24x^3y
```

-- this output is the standard form of the product `M*F`.

A.1.3. `WMul(F,G)`

Purpose: computes the product `F*G` of the Weyl polynomials `F` and `G`.

Syntax `Weyl.WMul(F:POLY,G:POLY):POLY`

Input Two Weyl polynomials `F` and `G`.

Output A polynomial which is the standard form of the product `F*G`.

Example Consider the Weyl algebra $A_2 = \mathbb{Z}_{101}[x_1,x_2,y_1,y_2]$, then we can perform multiplication of various polynomials in A_2 as follows:

```
A2::=ZZ/(101)[x[1..2],y[1..2]]; -- Define the appropriate ring
Use A2;
Weyl.WMul(x[1]^11,y[1]^11);
x[1]^11y[1]^11
```

−− this is the standard form of the product of $x_1^{11}, y_1^{11} \in A_2$

```
Weyl.WMul(y[1]^11,x[1]^11);
x[1]^11y[1]^11+20x[1]^10y[1]^10-10x[1]^9y[1]^9+33x[1]^8y[1]
^8+23x[1]^7y[1]^7-17x[1]^6y[1]^6-x[1]^5y[1]^5- 18x[1]^4y[1]
^4-36x[1]^3y[1]^3-36x[1]^2y[1]^2+26x[1]y[1]-16
```

−− this is the standard form of the product of $y_1^{11}, x_1^{11} \in A_2$

```
F:=3x[1]^2y[1]^3-2x[2]y[2]^2+5x[2]-5y[2]-7;
G:=4x[1]^2y[1]^2-9x[2]y[2]-7x[1]+y[1]+11;
12x[1]^4y[1]^5 - 29x[1]^3y[1]^4 - 27x[1]^2x[2]y[1]^3y[2] -
8x[1]^2x[2]y[1]^2y[2]^2 - 21x[1]^3y[1]^3 + 3x[1]^2y[1]^4 +
20x[1]^2x[2]y[1]^2 + 4x[1]^2y[1]^3 - 20x[1]^2y[1]^2y[2] +
18x[2]^2y[2]^3 + 10x[1]^2y[1]^2 + 14x[1]x[2]y[2]^2 -
2x[2]y[1]y[2]^2 - 45x[2]^2y[2] - 42x[2]y[2]^2 - 35x[1]x[2]
+ 5x[2]y[1] + 35x[1]y[2] - 38x[2]y[2] - 5y[1]y[2] +
49x[1] - 46x[2] - 7y[1] - 10y[2] + 24
```

−− this is the standard form of the product `F*G` of polynomials `F` and `G`.

A.1.4. `WMult(F,G)`

Purpose: Just like the function explained in A.1.3, this function also computes the product `F*G` of the Weyl polynomials `F` and `G`. The only difference is that it is implemented in ApCoCoAServer for the faster computation while working with the Weyl polynomials of very large size. This will also be useful for the computations in Weyl algebra by using ApCoCoALib. The ApCoCoAServer should be running for using this function.

Syntax `Weyl.WMult(F:POLY,G:POLY):POLY`

Input Two Weyl polynomials `F` and `G`.

Output A polynomial which is the standard form of the product `F*G`.

A.1.5. `WPower(F,N)`

Purpose: Computes the integer-power `N` of a Weyl polynomial `F`.

Syntax `Weyl.WPower(F:POLY,N:INT):POLY`

Input 1st parameter `F`, a Weyl polynomial.
2nd parameter `N`, a positive integer.

Output `F^N` as a Weyl polynomial.

Example For instance to compute $(xy^3 - xy + 1)^4$ in $A_1 = \mathbb{Q}[x,y]$, we proceed as follows:

`A1::=QQ[x,y]]; Use A1;` --Define and activate the appropriate ring

```
Weyl.WPower(xy^3-xy+1,4);
x^4y^12-4x^4y^10+18x^3y^11+6x^4y^8-56x^3y^9+87x^2y^10-4x^4y^6
+60x^3y^7-204x^2y^8+105xy^9+x^4y^4-24x^3y^5+148x^2y^6-
180xy^7+2x^3y^3-32x^2y^4+84xy^5+x^2y^2-8xy^3-xy+1
```

-- this is the standard form of $(xy^3 - xy + 1)^4$.

A.1.6. `WNR(F,G)`

Purpose: Computes the normal remainder of a Weyl polynomial `F` with respect to a polynomial `G` or a set of polynomials in the list `G`. If `G` is a Gröbner basis then this function is used for the ideal membership problem. *The* ApCoCoAServer *should be running for using this function.*

Syntax

```
Weyl.WNR(F:POLY,G:POLY):POLY
Weyl.WNR(F:POLY,G:LIST):POLY
```

Input 1st parameter `F`, a Weyl polynomial.

2nd parameter `G`, a list of Weyl polynomials or simply a Weyl polynomial.

Output The normal remainder of `F` with respect to the tuple of the Weyl polynomials given by the list `G` using the normal remainder algorithm 2.3.18.

Example Consider the Weyl algebra $A_3 = \mathbb{Z}_7[x_1,x_2,x_3,\partial_1,\partial_2,\partial_3]$ with the term ordering σ=`DegRevLex`. Let $f_1 = -\partial_1^3\partial_2^5\partial_3^5 + x_2^5$, $f_2 = -3x_2\partial_2^5\partial_3^5 + x_2\partial_1^3$, $f_3 = -2\partial_1^4\partial_2^5 - x_1\partial_2^7 + x_3^3\partial_3^5$, and $f_4 = -\partial_1^3\partial_2^7\partial_3^6 + x_2^5$ be the given Weyl polynomials. To compute the normal remainder of f_1 with respect to $\mathscr{G} = (f_2, f_3)$ we proceed as follows

`A3::=ZZ/(7)[x[1..3],d[1..3]];` --`DegRevLex` is the default term ordering in ApCoCoA.

-- Define the appropriate ring using `d[1],d[2],d[3]` for the indeterminates $\partial_1,\partial_2,\partial_3$ respectively.

```
Use A3;
F1:=-d[1]^3d[2]^5d[3]^5+x[2]^5;
F2:=-3x[2]d[2]^5d[3]^5+x[2]d[1]^3;
```

```
F3:=-2d[1]^4d[2]^5-x[1]d[2]^7+x[3]^3d[3]^5;
F4:=x[2]^5-d[1]^3d[2]^7d[3]^6;
G:=[F2,F3];
Weyl.WNR(F1,G);
-d[1]^3d[2]^5d[3]^5 + x[2]^5
```

-- This is the normal remainder $\mathrm{NR}_{\sigma,\mathscr{G}}(f_1)$. Similarly, to compute $\mathrm{NR}_{\sigma,f_1}(f_4)$, run the following command:

```
Weyl.WNR(x[2]^5-d[1]^3d[2]^7d[3]^6,F1);
-x[2]^5d[2]^2d[3] - 3x[2]^4d[2]d[3] + x[2]^5 + x[2]^3d[3]
```

-- this output is the result of $\mathrm{NR}_{\sigma,f_1}(f_4)$.

A.1.7. `WSPoly(F,G)`

Purpose: Computes the S-polynomial of Weyl polynomials `F` and`G`.

Syntax `Weyl.WSPoly(F:POLY,G:POLY):POLY`

Input Both parameters `F` and `G` are Weyl polynomials.

Output The S-polynomial of `F` and `G`.

Example In the Weyl algebra A_3 of A.1.6, consider again the polynomials f_1, f_2 and f_3. For computing the S-polynomials (see Definition 2.3.23) $S_{f_1 f_2}$, $S_{f_2 f_3}$ using ApCoCoA, as before first define and activate the appropriate ring and the run the following commands:

```
F1:=-d[1]^3d[2]^5d[3]^5+x[2]^5;
F2:=-3x[2]d[2]^5d[3]^5+x[2]d[1]^3;
F3:=-2d[1]^4d[2]^5-x[1]d[2]^7+x[3]^3d[3]^5;
Weyl.WSPoly(F1,F2);
x[2]d[1]^6 - 3x[2]^6
Weyl.WSPoly(F2,F3);
-3x[1]x[2]d[2]^7d[3]^5 + 3x[2]x[3]^3d[3]^10 + 3x[2]x[3]^2d[3]
^9 - 2x[2]x[3]d[3]^8 - 2x[2]d[1]^7 - 2x[2]d[3]^7
-------------------------------
```

A.1.8. `WGB(...)`

Purpose: This function computes the Gröbner basis of the ideal `I` using the corresponding implementation in CoCoALib. *The* ApCoCoAServer *should be running in order to use this function.*

Syntax `Weyl.WGB(I:IDEAL,L:LIST,N:INT):LIST`

Input

1. Ideal `I` of A_n.
2. (optional) List `L` of positive integers corresponding to the numbers of the indeterminates that are to be eliminated while computing Gröbner basis of `I`.
3. (optional) Integer `N` $= 0$ or 1.

Output The list of Weyl polynomials forming the Gröbner basis of the ideal `I`. If the 2nd parameter is given as a list of positive integers, then the function returns the Gröbner basis computed with by eliminating the indeterminates corresponding to the positive integers in the list `L`. The default value for the list `L` is the empty list `[]`. If the value 0 is used for the 3rd parameter `N`, then the function will returns the complete Gröbner basis computed by the Weyl code implemented in the CoCoALib without reduction otherwise default value of 1 will be used for `N` and output will be the reduced Gröbner basis of `I`. Note that, user can interchange the position of the two optional 2nd and 3rd parameters.

Example Following commands illustrate how one can use this function for computing a left Gröbner basis of an ideal I of A_n

```
A1::=QQ[x,d]; -- Define the appropriate ring
Use A1;
I:=Ideal(x,d);
Weyl.WGB(I);
[1] -- Note that the Gröbner basis obtained is minimal.
-------------------------------
Weyl.WGB(I,0);
[x,y,1] -- The Gröbner basis obtained is not minimal.
-------------------------------
W3::=ZZ/(7)[x[1..3],y[1..3]];
Use W3;
I3:=Ideal(x[1]^3y[2],x[2]y[1]^2);
Set Indentation;
Weyl.WGB(I3,0);
```

```
[
x[2]y[1]^2,
x[1]^3y[2],
x[1]^3y[1]^2 + x[1]^2x[2]y[1]y[2] + x[1]x[2]y[2],
x[1]^2x[2]y[1]y[2]^2 + 2x[1]^2y[1]y[2] + x[1]x[2]y[2]^2 +
2x[1]y[2],
x[1]^2x[2]^2y[1]y[2] + x[1]x[2]^2y[2],
x[1]x[2]y[1]y[2]^2 + 2x[1]y[1]y[2] - 2x[2]y[2]^2 + 3y[2],
x[1]^2x[2]y[2]^2 + 2x[1]^2y[2],
x[1]x[2]^2y[1]y[2] - 2x[2]^2y[2],
x[1]^2x[2]^2y[2],
x[2]y[1]y[2]^2 + 2y[1]y[2],
x[1]x[2]y[2]^2 + 2x[1]y[2],
x[2]^2y[1]y[2],
x[1]x[2]^2y[2],
x[2]y[2]^2 + 2y[2],
x[2]^2y[2]]
-------------------------------
Weyl.WGB(I3); -- now the reduced Gröbner basis will be returned
[
x[2]^2y[2],
x[2]y[2]^2 + 2y[2],
x[1]^3y[1]^2 + x[1]^2x[2]y[1]y[2] + x[1]x[2]y[2],
x[1]^3y[2],
x[2]y[1]^2]
-------------------------------
Unset Indentation;
```

A.1.9. `TwoWGB(I)`

Purpose: Computes the two-sided σ-Gröbner basis `G` of a two-sided ideal `I` of A_n. Recall that the Weyl algebra A_n is *simple* when K is a field of characteristic 0. The usage of this function makes sense only when K has positive characteristic. *The* ApCoCoAServer *should be running for using this function.*

Syntax `Weyl.TwoWGB(I:IDEAL):LIST`

Input An ideal `I` of A_n.

Output The two-sided Gröbner basis of the ideal `I` as a list of Weyl polynomials.

Example We illustrate the usage of this function by the following ApCoCoA commands.

```
A2::=ZZ/(2)[x[1..2],y[1..2]];--Define the appropriate ring
Use A2;
Weyl.TwoWGB(Ideal(x[1],y[1]));
[1]
Weyl.TwoWGB(Ideal(x[1]^2+1,y[2]^2));
[x[1]^2+1,y[2]^2]
Weyl.TwoWGB(Ideal(x[1]^2-1,y[1]^2-x[1]));
[1]
Weyl.TwoWGB(Ideal(x[1]^2y[1]^2-x[2]^2+1,x[2]^2y[1]^2-1));
[x[2]^4 + x[1]^2 + x[2]^2, x[1]^2y[1]^2 + x[2]^2 + 1, x[2]
^2y[1]^2 + 1]
-------------------------------
```

A.1.10. `WDim(I)`

Purpose: Computes dimension (GK-dimension) of an ideal `I` of A_n. *The* ApCoCoAServer *should be running in order to use this function.*

Syntax `Weyl.WDim(I:IDEAL):INT`

Input Ideal `I` of a Weyl Algebra A_n.

Output An integer `N`, the GK-dimension of the ideal `I`.

Example The following commands illustrate the usage of this function.

```
A2::=QQ[x[1..2],y[1..2]];
Use A3;
I1:=Ideal(x[1]y[1] + 2x[2]y[2] - 5, y[1]^2 - y[2]);
Weyl.WDim(I);
2 -- this output is the GK-dimension of the ideal I.
I2:=Ideal(x[1]y[1] + 2x[2]y[2] - 5, y[1]^2 - y[2]-1);
Weyl.WDim(I);
```

`-1` -- if the dimension is zero then -1 will be returned

```
Use W2::=ZZ/(2)[x[1..2],y[1..2]]; --Define and activate W2
I3:=Ideal(y[2]^2 + 2x[2]^2y[2]^4 - 5, y[1]^2 - y[2]^2-y[1]
^2y[1]^2, x[2]^4-1);
```

`Weyl.WDim(I); 1` -- the dimension of `I3` in `W2` is 1.

A.1.11. `IsHolonomic(I)`

Purpose: Checks whether an ideal `I` of A_n is holonomic or not. Recall that an ideal `I` is said to be holonomic if and only if its dimension is n, the index of the Weyl algebra A_n.
The ApCoCoAServer *should be running in order to use this function.*

Syntax `Weyl.IsHolonomic(I:IDEAL):BOOL`

Input An ideal `I` of A_n.

Output `True`, if `I` is holonomic and `False` otherwise.

Example We explain the usage by the following commands:

`A2::=QQ[x[1..2],y[1..2]];` --Define the appropriate ring

```
Use A2;
I:=Ideal(x[1]y[1] + 2x[2]y[2] - 5, y[1]^2 - y[2]-1);
Weyl.IsHolonomic(I);
False
I:=Ideal(x[1]y[1] + 2x[2]y[2] - 5, y[1]^2 - y[2]^3-y[1]^2x[1]);
Weyl.IsHolonomic(I);
```

`True` -- the ideal `I` is holonomic.

A.1.12. `WRGB(G)`

Purpose: Converts a Gröbner basis `G` into the reduced Gröbner Basis. If `G` is not a Gröbner basis then the output will not be the reduced Gröbner basis.

Syntax `Weyl.WRGB(G:LIST):LIST`

Input A list `G`, of Weyl polynomials.

Output A reduced list `L` of Weyl polynomials such that $\langle \mathrm{L} \rangle = \langle \mathrm{G} \rangle$.

Example For instance, consider the Weyl algebra A_1 in indeterminates `x` and `d` over the field $\mathbb{Q}$ and let I be the ideal of A_1 generated by elements in the list `L:=[x,d,1]`. Then `L` is a Gröbner basis of I and its reduced Gröbner basis can be computed as follows:

```
A1::=QQ[x,d]; -- Define the appropriate ring
Use A1;
L:=[x,y,1];
Weyl.WRGB(L);
[1] -- this output is the reduced Gröbner basis of I.
```

A.1.13. `WLT(I)`

Purpose: Computes the leading term ideal of an ideal `I` of A_n.
The ApCoCoAServer *should be running in order to use this function.*

Syntax `Weyl.WLT(I:IDEAL):IDEAL`

Input An ideal `I` of A_n.

Output An ideal, which is the leading term ideal of `I`

Example

```
A2::=QQ[x[1..2],y[1..2]]; -- Define the appropriate ring
Use A2;
I:=Ideal(x[1]y[2],x[2]y[1]);
Weyl.WLT(I);
Ideal(x[2]^2y[2], x[2]y[2]^2, x[1]y[1], x[2]y[1], x[1]y[2])
-- this output is the leading terms ideal of I.
```

Many other functions have also been implemented for the package `Weyl`. These functions are not relevant to the results presented in this thesis and therefore we have not described them here. For instance, one can also use this package for computing the *characteristic ideal*, the *annihilating ideal* of a polynomial f^s using the algorithm of *Oaku* and *Takayama*, the *Bernstein-Sato polynomial* of a polynomial f. The detailed description of these functions is available on-line at WWW page:

`http://www.apcocoa.org/wiki?title=Category:Package_weyl`

or also from the 'help menu' of your installed ApCoCoA.

Appendix B

Implementation

B.1 Linear Algebra Attack (commutative)

```
Define LAA(PK, C, Dm)    DegC := Deg(C);
   NPi := Len(PK); -- no.  of public polynomials
   HF := NewList(NPi, 1);
   MC := Monomials(C);
   D1 := Deg(PK[1]); D2 := Deg(PK[2]);
   DegPi := [Deg(P)|P In PK];
   DegH := DegC - Max(DegPi);
   S := Sum(Indets());
   SH1 := 0; SH2 := 0; SC2 := 0; M2 := 0;
   S := Sum(Indets());
   For N := 0 To Dm Do
      M2 := M2 + DensePoly(N);
   EndFor;
   M2 := Support(M2);
   Sol := Mat([[]]);
   While Sol=Mat([[ ]]) Do
      MC := Monomials(C);
      For N := 0 To DegH Do
         SH1 := SH1 + DensePoly(N);
      EndFor;
      SH1 := Support(SH1);
      For N := 0 To DegC Do
         SC2 := SC2 + DensePoly(N);
      EndFor;
      SC2 := Support(SC2);
      SizeH := Len(SH1);
      While Len(MC) <> Len(SC2) Do
         Append(MC, Poly(1));
      EndWhile;
```

```
For I := 1 To Len(SC2) Do
   If LPP(SC2[I]) <> LPP(MC[I]) Then
      Insert(MC, I, 0);
      Remove(MC, Len(MC));
   EndIf;
EndFor;
MatB := Transposed(Mat([[LC(Term)|Term In MC]]));
NRows := Len(SC2);
Lis := [];       MonomPi := [];
For I := 1 To Len(PK) Do
   Append(Lis, SH1); -- Lis is list of general li's in ∑l_i p_i
   Append(MonomPi, Monomials(PK[I]));
EndFor;
Cols := ConcatLists([ConcatLists(Lis), M2]);
NCols := Len(Cols);
PrintLn(" Size of the Linear system = ",NRows," × ",NCols);
PrintLn("Creating matrix of coefficients .  .  .  ");
Ax := NewMat(NRows, NCols, 0);
For I := 1 To SizeH Do
   For K := 0 To (NPi-1) Do
      HF[K + 1] := Monomials(Cols[I + K*SizeH] * PK[K + 1]);
      While HF[K + 1] <> [] Do
         For J := 1 To NRows Do
            If Len(HF[K + 1]) = 0 Then Break;EndIf;
            Lpp := LPP(HF[K + 1][1]);
            If Lpp = SC2[J] Then
               Ax[J][I + K*SizeH] := LC(HF[K + 1][1]);
               Remove(HF[K + 1],1);
            EndIf;
         EndFor;
      EndWhile;
   EndFor;
   Print(".");
EndFor;
PrintLn();
I := NPi*SizeH + 1;
For J := 1 To NRows Do
   If Cols[I] = SC2[J] Then
      Ax[J][I] := 1;
      I := I + 1;
   EndIf;
EndFor;
PrintLn("Now trying to solve using LinBox ...");
Sol := $apcocoa/linbox.Solve(Ax,MatB);
If Sol = Mat([[]]) Then
   PrintLn("Increasing Degree of Li >>>>>>>>>>>");
   DegH := DegH + 1; SH1 := 0; SC2 := 0;
   HF := NewList(NPi,1);
   DegC := DegH + Max(DegPi);
```

```
      EndIf;
   EndWhile;
   NewLis := [];
   S2 := ConcatLists(List(Sol));
   For I := 1 To NPi Do
      Li := [Cols[J]|J In 1..SizeH];
      CLi := [S2[J]|J In ((I-1)*SizeH + 1)..(I*SizeH)];
      Append(NewLis, ScalarProduct(CLi, Li));
   EndFor;
   CM := [S2[J]|J In (NPi*SizeH + 1)..Len(S2)];
   C2 := 0;
   For I := 1 To NPi Do
      C2 := C2 + NewLis[I]*PK[I];
   EndFor;
   M2 := ScalarProduct(CM, M2);
   PrintLn("Message was = ", M2);
   Return [M2, NewLis];
EndDefine; -- EndOf LAA( )
```

B.2 Intelligent Linear Algebra Attack

```
Define ILAA(PK, C, Dm)
   -- PK is list of public polynomials
   -- C is ciphertext.
   -- Dm is degree of message polynomial M
   DegC := Deg(C);
   SizeC := Len(C);
   NPi := Len(PK); -- no.  of public polynomials
   HF := NewList(NPi,1);
   MC := Monomials(C);
   Inds := NumIndets();
   DegPi := [Deg(P)|P In PK];
   DegH := DegC-Max(DegPi);
   PrintLn("Initalizing degree Li --> ",DegH);
   S := Sum(Indets());
   SH1 := 0; SH2 := 0; SC2 := 0; M2 := 0;
   NewRing ::= QQ[x[1..NumIndets()]];
   For N := 0 To Dm Do
      M2 := M2 + DensePoly(N);
   EndFor;
   M2 := Support(M2);
   Using NewRing Do
      SH2 := CreateD(ZPQ(PK),ZPQ(C));
   EndUsing;
   Sol := Mat([[]]);
   SH2 := QZP(SH2);
   While Sol=Mat([[ ]]) Do
      SH1 := [];
```

```
Foreach MonH In SH2 Do
   If Deg(MonH)<=DegH Then
      Append(SH1,MonH);
   EndIf;
EndForeach;
PrintLn(" # SH1 = ", Len(SH1));
SizeH := Len(SH1);--SH2 := [];
Lis := []; MonomPi := [];
For I := 1 To NPi Do
   Append(Lis,SH1);
   -- Lis is list of general li's in encryption
   Append(MonomPi,Monomials(PK[I]));
EndFor;
WExpecC := []; Counter := 0;
WSC2 := [];
Foreach MonH In SH1 Do
   For I := 1 To NPi Do
      Append(WExpecC,MonH*PK[I]);
   EndFor;
   Counter := Counter + 1;
   If Mod(Counter,2000)=0 Then
      Using NewRing Do
         WExpecC := ZPQ(WExpecC);
         Append(WSC2,Sum(WExpecC));
         WSC2 := [Sum(WSC2)];
      EndUsing;
      WExpecC := [];
   EndIf;
EndForeach;
PrintLn();
Using NewRing Do
   WExpecC := ZPQ(WExpecC);
   Append(WSC2,Sum(WExpecC));
   WSC2 := Support(Sum(WSC2));
EndUsing;
WSC2 := QZP(WSC2);
PrintLn("................# WExpecC = ",Len(WSC2));
SC2 := WSC2;
SizeSC2 := Len(SC2);
SupC := Support(C);
CoefC := Coefficients(C);
CoefC := [Cast(Coef, INT) | Coef In CoefC];
MC2 := [];
For I :=1 To SizeSC2 Do
   If Len(SupC) > 0 Then
      If SupC[1]= SC2[I] Then
         Append(MC2,[CoefC[1]]);
         Remove(SupC,1);Remove(CoefC,1);
      Else
```

```
            Append(MC2,[Zero]);
          EndIf;
        Else
          Append(MC2,[Zero]);
        EndIf;
      EndFor;
      MatB := Mat(MC2);
      PrintLn("Calculating Ax .  .  .  .  .  ");
      NRows := Len(SC2);
      Cols := ConcatLists([ConcatLists(Lis),M2]);
      NCols := Len(Cols);
      PrintLn(" Dimension of Ax = ",NRows," X ",NCols);
      Ax := NewMat(NRows,NCols,0);
      For I := 1 To SizeH Do
        For K := 0 To (NPi-1) Do
          HF[K + 1] := Monomials(Cols[I + K*SizeH] * PK[K + 1]);
          While HF[K + 1]<>[] Do
            For J := 1 To NRows Do
              If Len(HF[K + 1])=0 Then Break; EndIf;
              Lpp := LPP(HF[K + 1][1]);
              If Lpp=SC2[J] Then
                Ax[J][I + K*SizeH] := LC(HF[K + 1][1]);
                Remove(HF[K + 1],1);
              EndIf;
            EndFor;
          EndWhile;
        EndFor;
        Print(".");
      EndFor;
      PrintLn();
      I := NPi*SizeH + 1;
      For J := 1 To NRows Do
        If Cols[I]=SC2[J] Then
          Ax[J][I] := Unit;
          I := I + 1;
        EndIf;
      EndFor;
      PrintLn(" System's size = ", NRows, " × ", NCols);
      PrintLn("Now trying to solve LinBox .  .  .");
      Sol := $apcocoa/linbox.Solve(Ax,MatB);
      If Sol = Mat([[]])
        OR NonZero(ConcatLists(List(Sol))) = [] Then
        PrintLn("Increasing Degree of Li >>>>>>>>>>>");
        DegH := DegH + 1; SH1 := 0; HF := NewList(NPi,1);
        SC2 := 0;MC := Monomials(C);Sol := Mat([[]]);
      EndIf;
    EndWhile;
    NewLis := [];
    S2 := ConcatLists(List(Sol));
```

```
   For I := 1 To NPi Do
      Li := [Cols[J]|J In 1..SizeH];
      CLi := [S2[J]|J In ((I-1)*SizeH + 1)..(I*SizeH)];
      Append(NewLis,ScalarProduct(CLi,Li));
   EndFor;
   CM := [S2[J]|J In (NPi*SizeH + 1)..Len(S2)];
   C2 := 0;
   For I := 1 To NPi Do
      C2 := C2 + NewLis[I]*PK[I];
   EndFor;
   M2 := ScalarProduct(CM,M2);--Message found
   C2 := C2 + M2;
   PrintLn("Message was = ", M2);
   PrintLn("CipherText = ", C2=C);
   Return [M2,NewLis,Sol];--,Cols,Ax,MatB];
EndDefine;--End of ILAA( )
```

B.3 Linear Algebra Attack for Weyl Algebras

```
Define WLAA(PK, C, Dm)    DegC := Deg(C);
   NPi := Len(PK); -- no.  of public polynomials
   HF := NewList(NPi, 1);
   MC := Monomials(C);
   D1 := Deg(PK[1]); D2 := Deg(PK[2]);
   DegPi := [Deg(P)|P In PK];
   DegH := DegC - Max(DegPi);
   S := Sum(Indets());
   SH1 := 0; SH2 := 0; SC2 := 0; M2 := 0;
   S := Sum(Indets());
   For N := 0 To Dm Do
      M2 := M2 + DensePoly(N);
   EndFor;
   M2 := Support(M2);
   For N := 0 To DegH Do
      SH1 := SH1 + DensePoly(N);
   EndFor;
   SH1 := Support(SH1);
   For N := 0 To DegC Do
      SC2 := SC2 + DensePoly(N);
   EndFor;
   SC2 := Support(SC2);
   Sol := Mat([[]]);
   While Sol=Mat([[ ]]) Do
      MC := Monomials(C);
      SizeH := Len(SH1);
      While Len(MC) <> Len(SC2) Do
         Append(MC, Poly(1));
```

```
EndWhile;
For I := 1 To Len(SC2) Do
   If LPP(SC2[I]) <> LPP(MC[I]) Then
      Insert(MC, I, 0);
      Remove(MC, Len(MC));
   EndIf;
EndFor;
MatB := Transposed(Mat([[LC(Term)|Term In MC]]));
NRows := Len(SC2);
Lis := [];       MonomPi := [];
For I := 1 To Len(PK) Do
   Append(Lis, SH1); -- Lis is list of general li's in ∑l_i p_i
   Append(MonomPi, Monomials(PK[I]));
EndFor;
Cols := ConcatLists([ConcatLists(Lis), M2]);
NCols := Len(Cols);
PrintLn(" Size of the Linear system = ",NRows," × ",NCols);
PrintLn("Creating matrix of coefficients .  .  .  ");
PrintLn("Time depends upon the size of the system ...  ");
Ax := NewMat(NRows, NCols, 0);
For I := 1 To SizeH Do
   For K := 0 To (NPi-1) Do
      HF[K + 1] := Monomials($apcocoa/weyl.WMul(Cols[I +
      K*SizeH], PK[K+1]));
      While HF[K + 1] <> [] Do
         For J := 1 To NRows Do
            If Len(HF[K + 1]) = 0 Then Break;EndIf;
            Lpp := LPP(HF[K + 1][1]);
            If Lpp = SC2[J] Then
               Ax[J][I + K*SizeH] := LC(HF[K + 1][1]);
               Remove(HF[K + 1],1);
            EndIf;
         EndFor;
      EndWhile;
   EndFor;
   Print(".");
EndFor;
PrintLn();
I := NPi*SizeH + 1;
For J := 1 To NRows Do
   If Cols[I] = SC2[J] Then
      Ax[J][I] := 1;
      I := I + 1;
   EndIf;
EndFor;
PrintLn("Now trying to solve using LinBox ...");
Sol := $apcocoa/linbox.Solve(Ax,MatB);
If Sol = Mat([[]]) Then
   PrintLn("Increasing Degree of Li >>>>>>>>>>>");
```

```
         DegH := DegH + 1;
         HF := NewList(NPi,1);
         SH1:= Support(Sum(SH1)+DensePoly(DegH));
         SC2:= Support(Sum(SC2)+DensePoly(DegC+1));
         DegC := DegH + Max(DegPi);
      EndIf;
   EndWhile;
   NewLis := [];
   S2 := ConcatLists(List(Sol));
   For I := 1 To NPi Do
      Li := [Cols[J]|J In 1..SizeH];
      CLi := [S2[J]|J In ((I-1)*SizeH + 1)..(I*SizeH)];
      Append(NewLis, ScalarProduct(CLi, Li));
   EndFor;
   CM := [S2[J]|J In (NPi*SizeH + 1)..Len(S2)];
   C2 := 0;
   For I := 1 To NPi Do
      C2 := C2 + $apcocoa/weyl.WMul(NewLis[I],PK[I]);
   EndFor;
   M2 := ScalarProduct(CM, M2);
   PrintLn("Message was = ", M2);
   Return [M2, NewLis];
EndDefine; -- EndOf WLAA( )
```

B.4 Intelligent Linear Algebra Attack for Weyl Algebras

```
Define WILAA(PK, C, Dm)
   -- PK is list of public polynomials
   -- C is ciphertext.
   -- Dm is degree of message polynomial M
   DegC := Deg(C);
   SizeC := Len(C);
   NPi := Len(PK); -- no.  of public polynomials
   HF := NewList(NPi,1);
   MC := Monomials(C);
   Inds := NumIndets();
   DegPi := [Deg(P)|P In PK];
   DegH := DegC-Max(DegPi);
   PrintLn("Initalizing degree Li --> ",DegH);
   S := Sum(Indets());
   SH1 := 0; SH2 := 0; SC2 := 0; M2 := 0;
   NewRing ::= QQ[x[1..NumIndets()]];
   For N := 0 To Dm Do
      M2 := M2 + DensePoly(N);
   EndFor;
   M2 := Support(M2);
```

```
Using NewRing Do
   SH2 := CreateD(ZPQ(PK),ZPQ(C));
   -- It is the set of candidate terms for Li's
EndUsing;
Sol := Mat([[]]);
SH2 := QZP(SH2);
While Sol=Mat([[ ]]) Do
   SH1 := [];
   Foreach MonH In SH2 Do
      If Deg(MonH)<=DegH Then
         Append(SH1,MonH);
      EndIf;
   EndForeach;
   PrintLn(" # SH1 = ", Len(SH1));
   SizeH := Len(SH1);--SH2 := [];
   Lis := []; MonomPi := [];
   For I := 1 To NPi Do
      Append(Lis,SH1);
      -- Lis is list of general li's in encryption
      Append(MonomPi,Monomials(PK[I]));
   EndFor;
   WExpecC := []; Counter := 0;
   WSC2 := [];
   Foreach MonH In SH1 Do
      For I := 1 To NPi Do
         Append(WExpecC,$apcocoa/weyl.WMul(MonH,PK[I]));
      EndFor;
      Counter := Counter + 1;
      If Mod(Counter,2000)=0 Then
         Using NewRing Do
            WExpecC := ZPQ(WExpecC);
            Append(WSC2,Sum(WExpecC));
            WSC2 := [Sum(WSC2)];
         EndUsing;
         WExpecC := [];
      EndIf;
   EndForeach;
   PrintLn();
   Using NewRing Do
      WExpecC := ZPQ(WExpecC);
      Append(WSC2,Sum(WExpecC));
      WSC2 := Support(Sum(WSC2));
   EndUsing;
   WSC2 := QZP(WSC2);
   PrintLn("................# WExpecC = ",Len(WSC2));
   SC2 := WSC2;
   SizeSC2 := Len(SC2);
   SupC := Support(C);
   CoefC := Coefficients(C);
```

```
CoefC := [Cast(Coef, INT) | Coef In CoefC];
MC2 := [];
For I :=1 To SizeSC2 Do
   If Len(SupC) > 0 Then
      If SupC[1]= SC2[I] Then
         Append(MC2,[CoefC[1]]);
         Remove(SupC,1);Remove(CoefC,1);
      Else
         Append(MC2,[0]);
      EndIf;
   Else
      Append(MC2,[0]);
   EndIf;
EndFor;
MatB := Mat(MC2);
PrintLn("Calculating Ax .   .   .   .   .   ");
NRows := Len(SC2);
Cols := ConcatLists([ConcatLists(Lis),M2]);
NCols := Len(Cols);
PrintLn(" Dimension of Ax = ",NRows," × ",NCols);
Ax := NewMat(NRows,NCols,0);
For I := 1 To SizeH Do
   For K := 0 To (NPi-1) Do
      HF[K + 1] := Monomials($apcocoa/weyl.WMul(Cols[I +
      K*SizeH], PK[K + 1]));
      While HF[K + 1]<>[] Do
         For J := 1 To NRows Do
            If Len(HF[K + 1])=0 Then Break; EndIf;
            Lpp := LPP(HF[K + 1][1]);
            If Lpp=SC2[J] Then
               Ax[J][I + K*SizeH] := LC(HF[K + 1][1]);
               Remove(HF[K + 1],1);
            EndIf;
         EndFor;
      EndWhile;
   EndFor;
   Print(".");
EndFor;
PrintLn();
I := NPi*SizeH + 1;
For J := 1 To NRows Do
   If Cols[I]=SC2[J] Then
      Ax[J][I] := 1;
      I := I + 1;
   EndIf;
EndFor;
PrintLn("Now trying to solve LinBox .   .   .");
Sol := $apcocoa/linbox.Solve(Ax,MatB);
If Sol = Mat([[]])
```

```
        OR NonZero(ConcatLists(List(Sol))) = [] Then
        PrintLn("Increasing Degree of Li >>>>>>>>>>>");
        DegH := DegH + 1; SH1 := 0; HF := NewList(NPi,1);
        SC2 := 0;MC := Monomials(C);Sol := Mat([[]]);
     EndIf;
  EndWhile;
  NewLis := [];
  S2 := ConcatLists(List(Sol));
  For I := 1 To NPi Do
     Li := [Cols[J]|J In 1..SizeH];
     CLi := [S2[J]|J In ((I-1)*SizeH + 1)..(I*SizeH)];
     Append(NewLis,ScalarProduct(CLi,Li));
  EndFor;
  CM := [S2[J]|J In (NPi*SizeH + 1)..Len(S2)];
  C2 := 0;
  For I := 1 To NPi Do
     C2 := C2 + $apcocoa/weyl.WMul(NewLis[I],PK[I]);
  EndFor;
  M2 := ScalarProduct(CM,M2);--Message found
  C2 := C2 + M2;
  PrintLn("Message was = ", M2);
  PrintLn("CipherText = ", C2=C);
  Return [M2,NewLis];
EndDefine;--End of WILAA( )
```

Appendix C

Examples Data

The aim of this Appendix is to include some data related to various examples that are presented in this thesis. For the reader's convenience, the title of each section below is the chapter number in which such examples are presented. In each section, there is an enumerated items list in which every item starts with the reference to the corresponding example in that chapter.

C.1 Chapter 2

(1) **Example 2.5.6** The Gröbner basis elements $g_1, \dots, g_7$ are:

$g_1 = x^6 + 2x^4\partial - 2x^3\partial^2 - 3x^2\partial^3 + 3x\partial^4 + \partial^5 + 3x^4 + x^3\partial - x^2\partial^2 - 3x\partial^3 - 3\partial^4 - x^3 - x^2\partial + x\partial^2 - x^2 - 2x\partial - \partial^2 - 2\partial + 2,$

$g_2 = \partial^6 + 3x^5 - 2x^4\partial - x^3\partial^2 - x^2\partial^3 + x\partial^4 + 3\partial^5 - 3x^4 - x^3\partial - 3x^2\partial^2 - x\partial^3 + 3\partial^4 - 3x^3 - x^2\partial - 2x\partial^2 + \partial^3 - 2x^2 - 2x\partial + x - \partial - 2$"

$g_3 = x\partial^5 + 2x^5 - x^4\partial - 3x^3\partial^2 + x^2\partial^3 + 2x\partial^4 + 3\partial^5 + x^4 - 2x^2\partial^2 - 3x^3 - 3x^2\partial + 3x\partial^2 + 2\partial^3 - 3x^2 + x\partial + 2\partial^2 + 3x - 2\partial + 1,$

$g_4 = x^5\partial - x^5 - 2x^4\partial + x^3\partial^2 + 2x^2\partial^3 + 3x\partial^4 + 3\partial^5 + x^4 + 2x^2\partial^2 + 3x\partial^3 - 3x^3 + 3x^2\partial - 3x\partial^2 + 2\partial^3 + x^2 - 2x\partial - x - 3\partial,$

$g_5 = x^4\partial^2 - 3x^5 - 3x^4\partial - 2x^3\partial^2 - 3x^2\partial^3 + 3\partial^5 - 3x^3\partial + 3x^2\partial^2 - 2x\partial^3 - 3\partial^4 - 2x^3 + 3x^2\partial - 3x^2 + 2x\partial - 2\partial^2 + x - 2$",

$g_6 = x^2\partial^4 + 3x^4\partial + 2x^3\partial^2 - x^2\partial^3 + x\partial^4 - 2\partial^5 + x^4 - \partial^4 + 2x^3 + x^2\partial + 3x\partial^2 - 2\partial^3 + 2x^2 + 2\partial^2 - 2x - 2\partial - 2,$

$g_7 = x^3\partial^3 + x^2\partial - \partial - 1$

C.2 Chapter 4

(1) **Example 4.1.3** The Gröbner basis G of the ideal $I = \langle f_1, f_2, f_3 \rangle$ consists of the following 26 Weyl polynomials in standard form:

$$
\begin{aligned}
G \quad = \quad & \{x_2\partial_3^5 + \partial_1^3, \quad x_2^5\partial_2 - x_2^4, \quad x_2^5\partial_1, \quad x_2^4\partial_1^3, \quad \partial_1^4\partial_2^5 - x_1\partial_2^7, \quad x_2^3\partial_1^6, \\
& x_1\partial_1^2\partial_2^8 - \partial_1\partial_2^8, \quad x_1x_2\partial_1^2\partial_2^7 + \partial_1^3\partial_2^3\partial_3^5 + \partial_1^6\partial_2^4 - x_2\partial_1\partial_2^7 + x_27, \\
& x_1x_2^4\partial_2^6 + x_2^3\partial_1^4\partial_2^3, \quad x_2^3\partial_1^4\partial_2^4 - x_1x_2^3\partial_2^6, \quad x_2^2\partial_1^6\partial_2^3 - x_2^{10}, \quad x_2^2\partial_1^9, \quad x_2^{11}, \\
& x_1\partial_2^6\partial_3^5 + x_1\partial_1^3\partial_2^7, \quad \partial_1^3\partial_2^4\partial_3^5 - x_1\partial_1^2\partial_2^7 + \partial_1\partial_2^7 - x_2^6 \\
& \partial_1^4\partial_2^3\partial_3^5 + \partial_1^7\partial_2^4 + x_1\partial_1^3\partial_2^6, \quad x_1^2\partial_2^{10} + \partial_1^3\partial_2^8, \\
& x_1^2x_2\partial_2^9 + x_2\partial_1^3\partial_2^7 - x_1^2\partial_2^8 - \partial_1^3\partial_2^6, \quad x_1^2x_2^2\partial_2^8 + x_2^2\partial_1^3\partial_2^6, \\
& x_1x_2^2\partial_1^3\partial_2^6, \quad x_1x_2^3\partial_1^2\partial_2^6 - x_23\partial_1\partial_2^6, \quad x_2\partial_1^7\partial_2^4 + x_1x_2\partial_1^3\partial_2^6 - \partial_1^7\partial_2^3, \\
& \partial_1^9\partial_2^3 + x_12x_2\partial_1\partial_2^8, \quad x_2\partial_1^{12}, \quad \partial_1^{15}, \partial_1^3\partial_2^3\partial_3^{10} + x_1^2\partial_1\partial_2^9\}
\end{aligned}
$$

(2) **Example 4.3.3**

The polynomials p_1 and p_2 of the public key Q are

$p_1 = -4x_1^{10}x_2^9\partial_1^{10}\partial_2^7 + 6x_1^8x_2^{11}\partial_1^8\partial_2^9 + 4x_1^{10}x_2^{10}\partial_1^{10}\partial_2^4 - 3x_1^{10}x_2^8\partial_1^{10}\partial_2^6 + 4x_1^9x_2^9\partial_1^9\partial_2^7 + x_1^8x_2^{10}\partial_1^8\partial_2^8 +$
$6x_1^{10}x_2^6\partial_1^{10}\partial_2^8 - 4x_1^7x_2^{11}\partial_1^7\partial_2^9 - 5x_1^{10}x_2^9\partial_1^{10}\partial_2^4 + 2x_1^{10}x_2^6\partial_1^{10}\partial_2^7 - x_1^{10}x_2^9\partial_1^{10}\partial_2^3 - 4x_1^9x_2^{10}\partial_1^9\partial_2^4 + 6x_1^{10}x_2^8\partial_1^{10}\partial_2^4 +$
$3x_1^9x_2^8\partial_1^9\partial_2^6 - x_1^{10}x_2^6\partial_1^{10}\partial_2^6 - 5x_1^7x_2^{10}\partial_1^7\partial_2^8 - 6x_1^9x_2^6\partial_1^9\partial_2^8 - 2x_1^6x_2^{11}\partial_1^6\partial_2^9 + 5x_1^{10}x_2^8\partial_1^{10}\partial_2^3 + 5x_1^9x_2^9\partial_1^9\partial_2^4 -$
$2x_1^9x_2^6\partial_1^9\partial_2^7 + 2x_1^{10}x_2^8\partial_1^{10}\partial_2^2 + x_1^9x_2^9\partial_1^9\partial_2^3 - 4x_1^{10}x_2^7\partial_1^{10}\partial_2^3 - 6x_1^9x_2^8\partial_1^9\partial_2^4 + x_1^9x_2^6\partial_1^9\partial_2^6 + 4x_1^6x_2^{10}\partial_1^6\partial_2^8 +$
$3x_1^5x_2^{11}\partial_1^5\partial_2^9 + 2x_1^{10}x_2^7\partial_1^{10}\partial_2^2 - 5x_1^9x_2^8\partial_1^9\partial_2^3 - x_1^6x_2^{11}\partial_1^3\partial_2^9 - 4x_1^3x_2^{11}\partial_1^6\partial_2^9 - 6x_1^{10}x_2^7\partial_1^{10}\partial_2 - 2x_1^9x_2^8\partial_1^9\partial_2^2 -$
$6x_1^{10}x_2^6\partial_1^{10}\partial_2^2 + 4x_1^9x_2^7\partial_1^9\partial_2^3 - 6x_1^5x_2^{10}\partial_1^5\partial_2^8 + 4x_1^5x_2^{11}\partial_1^3\partial_2^9 + 6x_1^4x_2^{11}\partial_1^4\partial_2^9 + 5x_1^3x_2^{11}\partial_1^5\partial_2^9 - 3x_1^{10}x_2^6\partial_1^{10}\partial_2 -$
$2x_1^9x_2^7\partial_1^9\partial_2^2 + 2x_1^6x_2^{10}\partial_1^3\partial_2^8 - 5x_1^3x_2^{10}\partial_1^6\partial_2^8 + 4x_1^5x_2^{11}\partial_1^2\partial_2^9 - 6x_1^4x_2^{11}\partial_1^3\partial_2^9 - 2x_1^3x_2^{11}\partial_1^4\partial_2^9 + 5x_1^{10}x_2^6\partial_1^{10} +$
$6x_1^9x_2^7\partial_1^9\partial_2 + 6x_1^9x_2^6\partial_1^9\partial_2^2 + 5x_1^5x_2^{10}\partial_1^3\partial_2^8 + x_1^4x_2^{10}\partial_1^4\partial_2^8 + 3x_1^3x_2^{10}\partial_1^5\partial_2^8 - 2x_1^4x_2^{11}\partial_1^2\partial_2^9 - x_1^3x_2^{11}\partial_1^3\partial_2^9 +$
$3x_1^9x_2^6\partial_1^9\partial_2 - 2x_1^8x_2^5\partial_1^7\partial_2^5 + 5x_1^5x_2^{10}\partial_1^2\partial_2^8 - x_1^4x_2^{10}\partial_1^3\partial_2^8 + 4x_1^3x_2^{10}\partial_1^4\partial_2^8 - 5x_1^4x_2^{11}\partial_1\partial_2^9 - 5x_1^3x_2^{11}\partial_1^2\partial_2^9 -$
$5x_1^9x_2^6\partial_1^9 + 4x_1^4x_2^{10}\partial_1^2\partial_2^8 + 2x_1^3x_2^{10}\partial_1^3\partial_2^8 - 2x_1^3x_2^{11}\partial_1\partial_2^9 + x_1^8x_2^4\partial_1^7\partial_2^4 - 4x_1^7x_2^4\partial_1^8\partial_2^4 + 4x_1^7x_2^5\partial_1^6\partial_2^5 -$
$2x_1^6x_2^5\partial_1^7\partial_2^5 - 3x_1^4x_2^{10}\partial_1\partial_2^8 - 3x_1^3x_2^{10}\partial_1^2\partial_2^8 - 6x_1^3x_2^{11}\partial_2^9 + 4x_1^3x_2^{10}\partial_1\partial_2^8 - 2x_1^5x_2^5\partial_1^6\partial_2^5 - x_1^3x_2^{10}\partial_2^8 +$
$6x_1^8x_2^4\partial_1^7 + 2x_1^7x_2^4\partial_1^8 - 4x_1^8\partial_1^7\partial_2^4 + 3x_1^7\partial_1^8\partial_2^4 - x_1^8x_2^3\partial_1^7 + 4x_1^7x_2^3\partial_1^8 + 3x_1^8\partial_1^7\partial_2^3 + x_1^7\partial_1^8\partial_2^3 - 4x_1^8x_2^2\partial_1^7 +$
$3x_1^7x_2^2\partial_1^8 + 4x_1^8x_2\partial_1^7\partial_2 - 3x_1^7x_2\partial_1^8\partial_2 + 5x_1^8\partial_1^7\partial_2^2 + 6x_1^7\partial_1^8\partial_2^2 - 5x_1^8\partial_1^7 + 5x_1^7x_2\partial_1^7 + 5x_1^7\partial_1^8 - 3x_1^7\partial_1^7\partial_2 -$
$5x_1^3x_2^5\partial_1^2\partial_2^5 - 5x_1^2x_2^5\partial_1^3\partial_2^5 - 5x_1^7\partial_1^7 + 6x_1^4x_2^5\partial_2^5 + 6x_1^3x_2^5\partial_1\partial_2^5 - 2x_1x_2^5\partial_1^3\partial_2^5 - 2x_2^5\partial_1^4\partial_2^5 - 3x_1^7\partial_1^6 -$
$6x_1^6x_2\partial_1^6 - 2x_1^6\partial_1^7 + x_1^6\partial_1^6\partial_2 + 2x_1^3x_2^5\partial_2^5 - 4x_1^2x_2^5\partial_1\partial_2^5 + 6x_1x_2^5\partial_1^2\partial_2^5 - 4x_2^5\partial_1^3\partial_2^5 + 6x_1^6\partial_1^6 + 2x_1^2x_2^5\partial_2^5 -$
$4x_1x_2^5\partial_1\partial_2^5 - x_2^5\partial_1^2\partial_2^5 - 2x_1^5\partial_1^6 + 6x_1x_2^5\partial_2^5 + 3x_2^6\partial_2^5 - 4x_2^5\partial_1\partial_2^5 - 5x_2^5\partial_2^6 - x_2^5\partial_2^4 - 6x_2^4\partial_2^5 - x_2^4\partial_2^4 -$
$2x_2^3\partial_2^4 - 6x_2^5 - 6x_1^3\partial_1^2 + x_1^2x_2\partial_1^2 - 5x_1^2\partial_1^3 - 3x_2^4\partial_2 + 2x_1^2\partial_1^2\partial_2 + 4x_2\partial_2^4 + 2\partial_2^5 + 2x_1^4 + 4x_1^3x_2 -$
$5x_2^4 + 6x_1^3\partial_1 - x_1^2\partial_1^2 - 5x_1\partial_1^3 + 3x_2\partial_1^3 - 2\partial_1^4 - 5x_1^3\partial_2 - 6x_2^3\partial_2 + 6\partial_1^3\partial_2 - 3x_2\partial_2^3 - 4\partial_2^4 + x_1^3 -$
$3x_1^2x_2 + 6x_2^3 - 4x_1x_2\partial_1 + 6x_2\partial_1^2 + 6\partial_1^3 - 6x_1^2\partial_2 - 2x_2^2\partial_2 + 5x_1\partial_1\partial_2 - \partial_1^2\partial_2 + 6x_2\partial_2^2 + \partial_2^3 - 6x_1^2 -$

$$2x_1x_2 - x_2^2 + 5x_1\partial_1 - 5x_2\partial_1 + 6\partial_1^2 - 4x_1\partial_2 - 4x_2\partial_2 + 3\partial_1\partial_2 - 5\partial_2^2 - 2x_1 + 2x_2 + \partial_1 - \partial_2 + 1,$$

and

$$p_2 = -5x_1^9x_2^{12}\partial_1^{13}\partial_2^{14} - 2x_1^7x_2^{14}\partial_1^{11}\partial_2^{16} + 5x_1^9x_2^{13}\partial_1^{13}\partial_2^{11} + 6x_1^9x_2^{11}\partial_1^{13}\partial_2^{13} + 4x_1^8x_2^{12}\partial_1^{12}\partial_2^{14} - x_1^7x_2^{13}\partial_1^{11}\partial_2^{15} +$$
$$x_1^9x_2^9\partial_1^{13}\partial_2^{15} - 2x_1^6x_2^{14}\partial_1^{10}\partial_2^{16} - 3x_1^9x_2^{12}\partial_1^{13}\partial_2^{11} - 4x_1^9x_2^9\partial_1^{13}\partial_2^{14} - x_1^9x_2^{12}\partial_1^{13}\partial_2^{10} - 4x_1^8x_2^{13}\partial_1^{12}\partial_2^{11} +$$
$$x_1^9x_2^{11}\partial_1^{13}\partial_2^{11} + 5x_1^9x_2^{10}\partial_1^{13}\partial_2^{12} + 3x_1^8x_2^{11}\partial_1^{12}\partial_2^{13} + 2x_1^9x_2^9\partial_1^{13}\partial_2^{13} - x_1^6x_2^{13}\partial_1^{10}\partial_2^{15} - 6x_1^8x_2^9\partial_1^{12}\partial_2^{15} +$$
$$3x_1^5x_2^{14}\partial_1^9\partial_2^{16} + 5x_1^9x_2^{11}\partial_1^{13}\partial_2^{10} + 5x_1^8x_2^{12}\partial_1^{12}\partial_2^{11} - 2x_1^8x_2^9\partial_1^{12}\partial_2^{14} - 2x_1^9x_2^{11}\partial_1^{13}\partial_2^9 + 6x_1^8x_2^{12}\partial_1^{12}\partial_2^{10} -$$
$$4x_1^9x_2^{10}\partial_1^{13}\partial_2^{10} - 6x_1^8x_2^{11}\partial_1^{12}\partial_2^{11} + x_1^9x_2^9\partial_1^{13}\partial_2^{11} - 4x_1^8x_2^{10}\partial_1^{12}\partial_2^{12} + x_1^8x_2^9\partial_1^{12}\partial_2^{13} - 5x_1^5x_2^{13}\partial_1^9\partial_2^{15} +$$
$$x_1^4x_2^{14}\partial_1^8\partial_2^{16} - 2x_1^9x_2^{10}\partial_1^{13}\partial_2^9 - 4x_1^8x_2^{11}\partial_1^{12}\partial_2^{10} + 2x_1^5x_2^{14}\partial_1^6\partial_2^{16} - 5x_1^2x_2^{14}\partial_1^9\partial_2^{16} + x_1^9x_2^{10}\partial_1^{13}\partial_2^8 - x_1^8x_2^{11}\partial_1^{12}\partial_2^9 +$$
$$6x_1^9x_2^9\partial_1^{13}\partial_2^9 - 2x_1^8x_2^{10}\partial_1^{12}\partial_2^{10} - 6x_1^8x_2^9\partial_1^{12}\partial_2^{11} - 6x_1^4x_2^{13}\partial_1^8\partial_2^{15} + 5x_1^4x_2^{14}\partial_1^6\partial_2^{16} - 5x_1^3x_2^{14}\partial_1^7\partial_2^{16} +$$
$$3x_1^2x_2^{14}\partial_1^8\partial_2^{16} - 6x_1^9x_2^9\partial_1^{13}\partial_2^8 - x_1^8x_2^{10}\partial_1^{12}\partial_2^9 + x_1^5x_2^{13}\partial_1^6\partial_2^{15} + 4x_1^2x_2^{13}\partial_1^9\partial_2^{15} - 3x_1^4x_2^{14}\partial_1^5\partial_2^{16} - x_1^3x_2^{14}\partial_1^6\partial_2^{16} +$$
$$4x_1^2x_2^{14}\partial_1^7\partial_2^{16} + 2x_1^9x_2^9\partial_1^{13}\partial_2^7 - 6x_1^8x_2^{10}\partial_1^{12}\partial_2^8 + 3x_1^8x_2^9\partial_1^{12}\partial_2^9 - 4x_1^4x_2^{13}\partial_1^6\partial_2^{15} + 4x_1^3x_2^{13}\partial_1^7\partial_2^{15} - 5x_1^2x_2^{13}\partial_1^8\partial_2^{15} -$$
$$5x_1^3x_2^{14}\partial_1^5\partial_2^{16} + 6x_1^2x_2^{14}\partial_1^6\partial_2^{16} - 3x_1^8x_2^9\partial_1^{12}\partial_2^8 + 5x_1^4x_2^{13}\partial_1^5\partial_2^{15} + 6x_1^3x_2^{13}\partial_1^6\partial_2^{15} + 2x_1^2x_2^{13}\partial_1^7\partial_2^{15} -$$
$$2x_1^3x_2^{14}\partial_1^4\partial_2^{16} - 6x_1^2x_2^{14}\partial_1^5\partial_2^{16} + x_1^8x_2^9\partial_1^{12}\partial_2^7 + 4x_1^3x_2^{13}\partial_1^5\partial_2^{15} + 3x_1^2x_2^{13}\partial_1^6\partial_2^{15} - 6x_1^2x_2^{14}\partial_1^4\partial_2^{16} - x_1^3x_2^{13}\partial_1^4\partial_2^{15} -$$
$$3x_1^2x_2^{13}\partial_1^5\partial_2^{15} + 6x_1^2x_2^{14}\partial_1^3\partial_2^{16} - 3x_1^2x_2^{13}\partial_1^4\partial_2^{15} + 3x_1^2x_2^{13}\partial_1^3\partial_2^{15} + 3x_1^8\partial_1^7 + x_1^7x_2\partial_1^7 + 2x_1^7\partial_1^7 - x_1^7\partial_1^6 +$$
$$4x_1^6x_2\partial_1^6 - 5x_1^6\partial_1^6 - 2x_2^5\partial_1\partial_2^5 + x_2^5\partial_2^6 + 4x_2^5\partial_2^5 + 5x_2^4\partial_1\partial_2^4 - 4x_2^4\partial_2^5 + 3x_2^4\partial_2^4 + 3x_2^3\partial_2^4 + 4x_2^4\partial_1 -$$
$$2x_1^3\partial_1^2 - 5x_1^2x_2\partial_1^2 - 2x_2^4\partial_2 + 6\partial_1\partial_2^4 - 3\partial_2^5 + 5x_1^4 + 6x_1^3x_2 + 5x_2^4 - 5x_2^3\partial_1 + 3x_1^2\partial_1^2 - 6x_1\partial_1^3 -$$
$$2x_2\partial_1^3 - 4x_2^3\partial_2 + 2\partial_1\partial_2^3 + 5x_1^3 + 2x_1^2x_2 + 2x_2^3 - 5x_1^2\partial_1 - 6x_1x_2\partial_1 + 6x_2^2\partial_1 + x_1\partial_1^2 - 4x_2\partial_1^2 -$$
$$4\partial_1^3 - 3x_2^2\partial_2 - 6x_2\partial_1\partial_2 + 3x_2\partial_2^2 - \partial_1\partial_2^2 + 3\partial_2^3 - 5x_1^2 - 3x_1x_2 + 2x_2^2 - 2x_1\partial_1 - x_2\partial_1 + 5\partial_1^2 -$$
$$x_2\partial_2 + 2\partial_2^2 - 4x_2 + 6\partial_1 - \partial_2 + 1.$$

(3) **Example 4.3.6**

The polynomials p_1, p_2, and p_2 of the public key Q are

$$p_1 = x_2^2\partial_1^8\partial_2^5\partial_3^5 + x_2\partial_1^5\partial_2^3\partial_3^{10} - \partial_1^3\partial_2^6\partial_3^{10} + x_2\partial_1^8\partial_2^4\partial_3^5 + \partial_1^3\partial_2^5\partial_3^{10} + x_1^7x_2^3\partial_2^7 - x_1^5x_2^5\partial_2^7 - x_2^2\partial_1^5\partial_2^5\partial_3^5 -$$
$$x_2\partial_2^5\partial_3^{10} + x_1^5x_2^4\partial_2^6 - x_1^2x_2^6\partial_1^2\partial_3^5 + x_2^8\partial_1^2\partial_3^5 - x_2\partial_1^5\partial_2^4\partial_3^5 + x_1^2x_2^3\partial_1^4\partial_2^5 - x_2^5\partial_1^4\partial_2^5 + \partial_2^4\partial_3^{10} - x_1^5\partial_1\partial_2^7 -$$
$$x_1x_2^6\partial_1\partial_3^5 - x_2^4\partial_1^4\partial_2^4 - x_1x_2^3\partial_1^3\partial_2^5 + x_1^4\partial_2^7 + x_2^6\partial_3^5 - x_1^2x_2^3\partial_2\partial_3^5 + x_2^5\partial_2\partial_3^5 + x_2^3\partial_1^4\partial_2^3 - \partial_1^5\partial_2^5 + x_1^2x_2^3\partial_3^5 -$$
$$x_2^5\partial_3^5 + x_2^3\partial_1^2\partial_3^5 + \partial_2^5\partial_3^5 - x_2^4\partial_3^5 - x_1^4x_2^3\partial_2 + x_1^2x_2^5\partial_2 + x_2\partial_2^2\partial_3^5 - x_2\partial_2\partial_3^5 - x_1^2x_2^4 + x_1^2x_2^3 - x_2^5 +$$
$$\partial_1^3\partial_2^2 + \partial_3^5 + x_1^2\partial_1\partial_2 - \partial_1^3\partial_2 - x_1\partial_2 - \partial_2^2 + \partial_2 - 1,$$

$$p_2 = -x_1^6x_2^7\partial_2 + x_1^2x_2^{11}\partial_2 - x_1^6x_2^5x_3^2\partial_2 + x_1^2x_2^9x_3^2\partial_2 - x_1^2x_3^2\partial_3^{10} - x_1^6x_2^2\partial_3^5 + x_1^4x_2^4\partial_3^5 + x_1^4x_2^2x_3^2\partial_3^5 -$$
$$x_2^6x_3^2\partial_3^5 - x_1^2x_3^2\partial_1^3\partial_2\partial_3^5 - x_1^2\partial_2\partial_3^{10} + x_1^6x_2^6 + x_1^6x_2^4x_3^2 - x_1^6x_2^2\partial_1^3\partial_2 - x_2^6x_3^2\partial_1^3\partial_2 + x_1^4x_2^2\partial_2\partial_3^5 - x_2^6\partial_2\partial_3^5 +$$
$$x_1^4x_2^6 - x_2^{10} + x_1^6x_2^3\partial_2 - x_1^4x_2^5\partial_2 - x_1^2x_2^3x_3^4\partial_2 + x_2^5x_3^4\partial_2 - x_1^4x_2\partial_3^5 - x_2^5\partial_3^5 + x_2^2x_3^2\partial_2\partial_3^5 + x_1^4x_2^5 - x_2^9 -$$
$$x_1^2x_2^6\partial_2 + x_1^4x_2^2x_3^2\partial_2 + x_1^2x_2^4x_3^2\partial_2 + x_1^4x_2^4 - x_2^4x_3^4 + x_1^2x_2^3\partial_1^3 + x_2x_3^2\partial_3^5 + x_1^6x_2 + x_1^2x_2^5 - x_1^4x_2x_3^2 + x_1^2x_2^3x_3^2 +$$
$$x_2^5 - x_1^4\partial_2 + x_3^4\partial_2 - \partial_3^5 - x_2^4 + x_1^2x_2\partial_2 - x_2x_3^2\partial_2 + \partial_1^3\partial_2 + x_1^2 - x_3^2 - \partial_2 - 1,$$ and

$p_3 = x_2^7x_3^4\partial_2^{12}\partial_3^5 - x_1^2x_2^9x_3^4\partial_2^{12} + x_2^{11}x_3^4\partial_2^{12} + x_1^3x_3^4\partial_1\partial_2^{12}\partial_3^5 + x_1x_2^2x_3^4\partial_1\partial_2^{12}\partial_3^5 - x_1^7x_3^4\partial_1\partial_2^{12} - x_1^5x_2^2x_3^4\partial_1\partial_2^{12} +$
$x_1x_2^6x_3^4\partial_1\partial_2^{12} + x_2^{10}\partial_1^7\partial_2^6 - x_2^7\partial_1^4\partial_2^7\partial_3^5 + x_2^2x_3^4\partial_2^{12}\partial_3^5 + x_1^2x_2^9\partial_1^4\partial_2^7 - x_2^{11}\partial_1^4\partial_2^7 - x_1^6x_3^4\partial_2^{12} + x_1^4x_2^2x_3^4\partial_2^{12} +$
$x_2^6x_3^4\partial_2^{12} + x_2^5x_3^4\partial_1\partial_2^6\partial_3^5 + x_1^4x_2^5x_3^4\partial_1\partial_2^6 + x_1^2x_2^7x_3^4\partial_1\partial_2^6 - x_2^9x_3^4\partial_1\partial_2^6 - x_1^2x_2^8\partial_1^4\partial_2^6 - x_2^{10}\partial_1^4\partial_2^6 - x_1x_2^9\partial_1^3\partial_2^7 -$
$x_1^4x_2^2\partial_1^7\partial_2^7 - x_1^2x_2^4\partial_1^7\partial_2^7 - x_2^6\partial_1^7\partial_2^7 - x_1^5x_2^4\partial_2^6\partial_3^5 + x_1x_2^8\partial_2^6\partial_3^5 + x_1^3x_2^5x_3^4\partial_2^6 - x_1x_2^7x_3^4\partial_2^6 - x_2^9x_3^2\partial_1\partial_2^6 -$
$x_1x_2^8\partial_1^3\partial_2^6 + x_1^2x_2^3\partial_1^7\partial_2^6 - x_2^5\partial_1^7\partial_2^6 - x_1^2x_2^7x_3^2\partial_2^7 - x_1^3x_2^2\partial_1^6\partial_2^7 + x_1x_2^4\partial_1^6\partial_2^7 + x_1^2x_2^4\partial_1\partial_2^6\partial_3^5 - x_1^4x_2^6\partial_1\partial_2^6 +$
$x_1^2x_2^8\partial_1\partial_2^6 + x_2^9x_3\partial_1\partial_2^6 + x_1x_2^7\partial_1^7\partial_2 - x_1^2x_2^6x_3^2\partial_2^6 - x_1x_2^3\partial_1^6\partial_2^6 + x_1^2x_2^7\partial_2^7 + x_1x_2^7x_3\partial_2^7 + x_1^2x_2^9\partial_3^5 -$
$x_2^{11}\partial_3^5 - x_3^4\partial_1\partial_2^6\partial_3^5 + x_1^3x_2^6\partial_2^6 - x_1^4x_3^4\partial_1\partial_2^6 + x_1^2x_2^2x_3^4\partial_1\partial_2^6 + x_1^4x_2\partial_1^4\partial_2^6 + x_1^2x_2^3\partial_1^4\partial_2^6 + x_1^3x_2\partial_2^6\partial_3^5 -$
$x_1x_2^6\partial_1^7 + x_2^7\partial_1^6\partial_2 + x_1^5x_2^3\partial_2^6 + x_1^3x_2^5\partial_2^6 + x_1^2x_2^6\partial_2^6 + x_1x_2^7\partial_2^6 + x_1x_2^6x_3\partial_2^6 - x_1^3x_3^4\partial_2^6 - x_1x_2^2x_3^4\partial_2^6 -$
$x_1^4x_3^2\partial_1\partial_2^6 - x_1^2x_2^2x_3^2\partial_1\partial_2^6 - x_2^4x_3^2\partial_1\partial_2^6 + x_1x_2^3\partial_1^3\partial_2^6 - x_1^4x_3^2\partial_2^7 - x_1^2x_2^2x_3^2\partial_2^7 - x_2^6\partial_1^6 + x_1^4x_3\partial_1\partial_2^6 +$
$x_1^2x_2^2x_3\partial_1\partial_2^6 + x_2^4x_3\partial_1\partial_2^6 - x_1x_2^5x_3^4\partial_1 - x_1x_2^6\partial_1^4 + x_1x_2^2\partial_1^7\partial_2 - x_1^3x_3^2\partial_2^6 + x_1^2x_2x_3^2\partial_2^6 + x_1x_2^2x_3^2\partial_2^6 +$
$x_1^4\partial_2^7 + x_1^2x_2^2\partial_2^7 + x_1^3x_3\partial_2^7 + x_1x_2^2x_3\partial_2^7 + x_1^2x_2^4\partial_3^5 + x_2^6\partial_3^5 - x_1^2x_2^8 + x_2^{10} + x_1^3x_3\partial_2^6 - x_1x_2^2x_3\partial_2^6 -$
$x_2^5x_3^4 + x_1x_2^5x_3^2\partial_1 + x_1x_2^5x_3^2\partial_2 + x_2^2\partial_1^6\partial_2 - x_1^3\partial_2^6 - x_1^2x_2\partial_2^6 - x_1x_2^2\partial_2^6 - x_1x_2x_3\partial_2^6 - x_1x_2^5x_3\partial_1 - x_1x_2^4x_3^2 +$
$x_2^5x_3^2 - x_1x_2^5\partial_2 - x_2^5x_3\partial_2 - x_2^5x_3 - x_1x_3^4\partial_1 - x_1x_2\partial_1^4 - x_2\partial_3^5 - x_1^2x_2^3 + x_1x_2^4 - x_2^5 + x_2^4x_3 - x_3^4 +$
$x_1x_3^2\partial_1 + x_1x_3^2\partial_2 - x_1x_3\partial_1 + x_3^2 - x_1\partial_2 - x_3\partial_2 - x_3 + 1$

(4) **Example 4.4.2**

The polynomials p_1, p_2, and p_2 of the public key Q are

$p_1 = -x_1^3x_2^6x_3^{10}\partial_1\partial_2^3 + 2x_1^3x_2^6x_3^7\partial_1\partial_2^3\partial_3^2 + x_1^2x_2^6\partial_1^3\partial_2^3\partial_3^4 + 3x_1x_3^{10}\partial_1\partial_2^2\partial_3^3 - x_1x_2^6\partial_1^3\partial_2^3\partial_3^4 + 3x_1^3x_2^3\partial_2^7\partial_3^4 +$
$3x_1^3x_2^6\partial_2^3\partial_3^4 + 3x_1^2x_2^6\partial_1\partial_2^3\partial_3^4 - 2x_1^3x_2^3\partial_2^6\partial_3^4 + x_1x_3^7\partial_1\partial_2^2\partial_3^5 - x_1x_3^9\partial_1\partial_2^2\partial_3^2 + x_1^3x_2^5\partial_2^3\partial_3^4 + 3x_1^2x_2^6\partial_2^3\partial_3^4 +$
$x_1x_2^6\partial_1\partial_2^3\partial_3^4 - x_1^3x_2^4\partial_2^4\partial_3^4 - 3x_1^3x_2^3\partial_2^5\partial_3^4 + 2x_1x_2x_3\partial_1^2\partial_2^6\partial_3^4 + x_3^{11}\partial_3^3 + 3x_1^3x_2^4\partial_2^3\partial_3^4 - 2x_1x_2^6\partial_2^3\partial_3^4 +$
$3x_2^6\partial_1\partial_2^3\partial_3^4 + 2x_1^3x_2^3\partial_2^4\partial_3^4 - x_1x_2x_3\partial_1^2\partial_2^5\partial_3^4 + x_1x_3^{10}\partial_1^2 + x_2x_3^{10}\partial_2^2 - x_3^{10}\partial_2^3 - 2x_1x_3^8\partial_1\partial_2^2\partial_3 - x_1^3x_2^4\partial_2^2\partial_3^4 -$
$3x_1x_2^3x_3\partial_1^2\partial_2^2\partial_3^4 - x_1^3x_2^3\partial_2^3\partial_3^4 + 3x_2^6\partial_2^3\partial_3^4 + 3x_1x_2^2x_3\partial_1^2\partial_2^3\partial_3^4 + 2x_1x_2x_3\partial_1^2\partial_2^4\partial_3^4 - 2x_3^8\partial_3^5 + 2x_1x_3^{10}\partial_1 +$
$x_1x_3^{10}\partial_2 + 2x_3^{10}\partial_3^2 - 2x_1x_3^7\partial_1^2\partial_3^2 - 2x_2x_3^7\partial_2^2\partial_3^2 + 2x_3^7\partial_2^3\partial_3^2 + 2x_1^3x_2^3\partial_2^2\partial_3^4 - 2x_1x_2^2x_3\partial_1^2\partial_2^2\partial_3^4 + x_1x_2x_3\partial_1^2\partial_2^3\partial_3^4 +$
$3x_3^{10}\partial_1 - 3x_1x_3^7\partial_1\partial_2^2 + \partial_1^3\partial_2^7\partial_3 + 3x_1x_3^7\partial_1\partial_3^2 - 2x_1x_3^7\partial_2\partial_3^2 - x_1^3x_2^3\partial_2\partial_3^4 + 2x_1x_2^2x_3\partial_1^2\partial_2\partial_3^4 - x_1^3\partial_2^4\partial_3^4 -$
$3x_3^9\partial_3 + 3\partial_1^3\partial_2^6\partial_3 + x_3^7\partial_1\partial_3^2 + 3x_1x_2x_3\partial_1^2\partial_2\partial_3^4 - 3x_1^3\partial_2^3\partial_3^4 - 2x_1x_2\partial_1^3\partial_2^3\partial_3 + 2x_2^2\partial_1^3\partial_2^3\partial_3 - 2x_2\partial_1^3\partial_2^4\partial_3 +$
$\partial_1^3\partial_2^5\partial_3 - 2x_1^3x_2^2\partial_3^4 + x_1x_2x_3\partial_1^2\partial_3^4 + 2x_1^3x_2\partial_2\partial_3^4 - x_1^3\partial_2^2\partial_3^4 + 3x_3\partial_2^4\partial_3^4 - x_3^8 + 3x_1x_2\partial_1^3\partial_2^3 - x_2\partial_1^3\partial_2^3\partial_3 -$
$3\partial_1^3\partial_2^4\partial_3 + x_1^3x_2\partial_3^4 + 3x_1^3\partial_2\partial_3^4 + 2x_3\partial_2^3\partial_3^4 - x_1^3\partial_2^4 - 3x_2^3\partial_2^4 - 2x_1x_3^2\partial_1^3\partial_3 - 2x_2\partial_1^3\partial_2^2\partial_3 + x_1^2x_2\partial_2^3\partial_3 +$
$x_1x_2\partial_1\partial_2^3\partial_3 - 2\partial_1^3\partial_2^3\partial_3 + 3x_1^3\partial_3^4 - x_2^2x_3\partial_3^4 + x_2x_3\partial_2\partial_3^4 + 3x_3\partial_2^2\partial_3^4 - 3x_1^3\partial_2^3 + 2x_1^2x_2\partial_2^3 - 2x_2^3\partial_2^3 +$
$2x_1x_2\partial_1\partial_2^3 + x_1x_2\partial_1^3\partial_3 - 3\partial_1^3\partial_2^2\partial_3 + 2x_1x_2\partial_2^3\partial_3 - x_2\partial_1\partial_2^3\partial_3 - 3x_2x_3\partial_3^4 - 2x_3\partial_2\partial_3^4 - 2x_1^3x_2^2 +$
$x_2^5 + 2x_1^3x_2\partial_2 - x_2^4\partial_2 - x_1^3\partial_2^2 - 3x_2^3\partial_2^2 - 3x_1x_2\partial_2^3 - 2x_2\partial_1\partial_2^3 + 3x_3\partial_2^4 + x_1^2x_3^2\partial_3 + x_1x_3^2\partial_1\partial_3 -$
$3x_1\partial_1^3\partial_3 - 2\partial_1^3\partial_2\partial_3 - x_2\partial_2^3\partial_3 - 2x_3\partial_3^4 + x_1^3x_2 + 3x_2^4 + 3x_1^3\partial_2 + 2x_2^3\partial_2 - 2x_2\partial_2^3 + 2x_3\partial_2^3 + 3x_1^2x_2\partial_3 +$
$2x_1x_3^2\partial_3 + 3x_1x_2\partial_1\partial_3 - x_3^2\partial_1\partial_3 + 3x_1^3 + 2x_2^3 - x_2^2x_3 + x_2x_3\partial_2 + 3x_3\partial_2^2 - 2x_1^2\partial_3 - x_1x_2\partial_3 - x_3^2\partial_3 -$
$2x_1\partial_1\partial_3 - 3x_2\partial_1\partial_3 - 3x_2x_3 - 2x_3\partial_2 + 3x_1\partial_3 - 3x_2\partial_3 + 2\partial_1\partial_3 - 2x_3 + 2\partial_3,$

$p_2 = 3x_1^2x_2^4x_3^9\partial_2^4\partial_3 + 2x_1^2x_2^2x_3^9\partial_2^6\partial_3 - 3x_1^2x_2^3x_3^7\partial_2^5\partial_3^3 + 3x_1^2x_2^2x_3^7\partial_2^6\partial_3^3 - x_1^2x_2^3x_3^9\partial_2^5 + x_1^2x_2^2x_3^9\partial_2^6 -$
$x_1^2x_2^4x_3^9\partial_2^3\partial_3 + x_1^2x_2^3x_3^9\partial_2^4\partial_3 + 2x_1^2x_2^2x_3^9\partial_2^5\partial_3 - x_1^2x_2^4x_3^7\partial_2^3\partial_3^3 + x_1^2x_2^3x_3^7\partial_2^4\partial_3^3 + x_1^2x_2^2x_3^7\partial_2^5\partial_3^3 + 2x_1^2x_2^4x_3^9\partial_2^3 -$
$3x_1^2x_2^4x_3^8\partial_2^4 - 2x_1^2x_2^3x_3^9\partial_2^4 - 2x_1^2x_2^2x_3^9\partial_2^5 - 2x_1^2x_2^2x_3^8\partial_2^6 + x_1^2x_2^2x_3^9\partial_2^4\partial_3 + x_1^2x_2^3x_3^7\partial_2^3\partial_3^3 - 2x_1^2x_2^2x_3^7\partial_2^4\partial_3^3 +$
$x_1^2x_2^4x_3^8\partial_2^3 - 2x_1^2x_2^3x_3^9\partial_2^3 - x_1^2x_2^3x_3^8\partial_2^4 - 3x_1^2x_2^2x_3^9\partial_2^4 - 2x_1^2x_2^2x_3^8\partial_2^5 +$
$x_1^2x_2^3x_3^9\partial_2^2\partial_3 + x_1^2x_2^2x_3^9\partial_2^3\partial_3 + x_1^2x_2^3x_3^7\partial_2^2\partial_3^3 - 2x_1^2x_2^2x_3^7\partial_2^3\partial_3^3 - 3x_1^3x_3^5\partial_1^2\partial_2^4\partial_3^3 - 2x_1x_3^5\partial_1^4\partial_2^4\partial_3^3 - 2x_1^2x_2^3x_3^9\partial_2^2 -$
$3x_1^2x_2^2x_3^9\partial_2^3 - x_1^2x_2^2x_3^8\partial_2^4 + 3x_1^2x_2^2x_3^7\partial_2^2\partial_3^3 + x_1x_2^4x_3^2\partial_1^4\partial_2^2\partial_3^3 - x_1x_2^3x_3^2\partial_1^4\partial_2^3\partial_3^3 - x_1^2x_3^5\partial_1^2\partial_2^4\partial_3^3 - x_1x_3^5\partial_1^3\partial_2^4\partial_3^3 -$
$x_1^2x_2^3x_3^8\partial_2^2 + x_1^2x_2^2x_3^9\partial_2^2 - x_1^2x_2^2x_3^8\partial_2^3 - 2x_1x_2^5x_3^2\partial_1^4\partial_3^3 + x_1^2x_2^2x_3^7\partial_2\partial_3^3 + 2x_1x_2^4x_3^2\partial_1^4\partial_2\partial_3^3 + 2x_1x_2^3x_3^2\partial_1^4\partial_2^2\partial_3^3 +$
$3x_1^2x_3^5\partial_1\partial_2^4\partial_3^3 - x_1x_3^5\partial_1^2\partial_2^4\partial_3^3 - 2x_3^5\partial_1^3\partial_2^4\partial_3^3 - 2x_1^2x_2^2x_3^9\partial_2 + 3x_1x_3^{10}\partial_3^3 - 3x_3^{11}\partial_3^3 + 2x_1x_2^4x_3^2\partial_1^4\partial_3^3 +$
$3x_1x_3^5\partial_1\partial_2^4\partial_3^3 - x_3^5\partial_1^2\partial_2^4\partial_3^3 + x_1^2x_3^9\partial_3^2 + 3x_3^9\partial_1\partial_2\partial_3^2 - 3x_1x_2^3x_3^2\partial_1^4\partial_3^3 + 3x_1x_3^5\partial_2^4\partial_3^3 - 2x_3^5\partial_1\partial_2^4\partial_3^3 +$
$2x_1x_3^9\partial_3^2 - 2x_3^{10}\partial_3^2 - 2x_3^5\partial_2^4\partial_3^3 + x_1^2x_3^8\partial_3 - 2x_2x_3^9\partial_3 + x_3^9\partial_2\partial_3 + 3x_3^8\partial_1\partial_2\partial_3 + 3x_1^6x_3^2\partial_1\partial_3^2 + 2x_1^4x_3^2\partial_1^3\partial_3^2 +$
$x_1^5x_3^2\partial_1\partial_3^2 + x_1^4x_3^2\partial_1^2\partial_3^2 - x_1^2x_3^7 + 2x_2x_3^8 - x_3^8\partial_2 - 3x_3^7\partial_1\partial_2 + 2x_1^5x_3^2\partial_3^2 + x_1^4x_3^2\partial_1\partial_3^2 - 2x_1x_2\partial_1\partial_2^4\partial_3^2 +$
$2x_1\partial_1\partial_2^5\partial_3^2 + 2x_1^4x_3^2\partial_3^2 - 3x_1x_2^2\partial_1\partial_2^2\partial_3^2 + 3x_1x_2\partial_1\partial_2^3\partial_3^2 + 3x_1\partial_1\partial_2^4\partial_3^2 + x_2^2x_3\partial_1^2\partial_2^2 - x_2x_3\partial_1^2\partial_2^3 +$
$x_1^3x_3^2\partial_3^2 + 3x_1x_2\partial_1\partial_2^2\partial_3^2 + 3x_1\partial_1\partial_2^3\partial_3^2 - 2x_2^3x_3\partial_1^2 + 2x_2^2x_3\partial_1^2\partial_2 + 2x_2x_3\partial_1^2\partial_2^2 - 3x_1^5\partial_3 - 2x_1^3\partial_1^2\partial_3 +$
$2x_1x_2\partial_1\partial_2\partial_3^2 - 2x_1\partial_1\partial_2^2\partial_3^2 + 2x_1^3x_2\partial_1 + 2x_2^2x_3\partial_1^2 - x_1x_2\partial_1^3 + x_1x_2\partial_1\partial_2^2 - x_1\partial_1\partial_2^3 - 2x_2\partial_2^4 +$
$2\partial_2^5 - x_1^4\partial_3 - x_1^3\partial_1\partial_3 - x_1\partial_1\partial_2\partial_3^2 - x_1^3x_3 + 3x_1^2x_2\partial_1 - 2x_1x_2^2\partial_1 + 3x_1x_2\partial_1^2 - 3x_1x_3\partial_1^2 - 3x_2x_3\partial_1^2 +$
$2x_1x_2\partial_1\partial_2 - 3x_2^2\partial_2^2 - 3x_2x_3\partial_2^2 + 2x_1\partial_1\partial_2^2 + 3x_2\partial_2^3 + 3x_3\partial_2^3 + 3\partial_2^4 - 3x_1^3\partial_3 + 2x_1^2\partial_1\partial_3 + x_1\partial_1^2\partial_3 -$
$x_2\partial_2^2\partial_3 + \partial_2^3\partial_3 + x_1\partial_1\partial_3^2 - x_1^2x_2 + 2x_1^2x_3 - x_2^2x_3 - 2x_1x_2\partial_1 + 2x_1x_3\partial_1 + x_2x_3\partial_2 + 3x_2\partial_2^2 + x_3\partial_2^2 +$
$3\partial_2^3 - 2x_1^2\partial_3 + 2x_2^2\partial_3 - 3x_1\partial_1\partial_3 - 2x_2\partial_2\partial_3 - 2\partial_2^2\partial_3 - x_1x_2 + 2x_1x_3 + x_2x_3 - 3x_1\partial_1 + 3x_3\partial_1 +$
$2x_2\partial_2 - 2\partial_2^2 - 3x_1\partial_3 - 2x_2\partial_3 - \partial_1\partial_3 + 3x_2 - \partial_2 - \partial_3 + 1$, and

$p_3 = -3x_1x_2^2x_3^{10}\partial_1^3\partial_2\partial_3 - 2x_1x_3^{10}\partial_1^3\partial_2^3\partial_3 + x_1x_2^2x_3^{10}\partial_1^3\partial_3 - x_1x_2x_3^{10}\partial_1^3\partial_2\partial_3 + 3x_1^2x_2x_3^{10}\partial_1\partial_2^2\partial_3 -$
$2x_1x_3^{10}\partial_1^3\partial_2^2\partial_3 - 3x_1^2x_3^{10}\partial_1\partial_2^3\partial_3 + x_1x_2^2x_3^8\partial_1^3\partial_3^3 - 2x_1x_2^2x_3^{10}\partial_1^3 - x_1x_2x_3^{10}\partial_1^3\partial_3 - x_1x_3^{10}\partial_1^3\partial_2\partial_3 +$
$2x_1^2x_2x_3^{10}\partial_2^2\partial_3 - x_1^2x_3^{10}\partial_1\partial_2^2\partial_3 + 3x_1x_2x_3^{10}\partial_1\partial_2^2\partial_3 - 2x_1^2x_3^{10}\partial_2^3\partial_3 - 3x_1x_3^{10}\partial_1\partial_2^3\partial_3 + x_1^2x_2^3x_3^9\partial_1 +$
$2x_1x_3^{10}\partial_1^3\partial_3 - 3x_1^2x_3^{10}\partial_2^2\partial_3 - 2x_1x_2x_3^{10}\partial_2^2\partial_3 - x_1x_3^{10}\partial_1\partial_2^2\partial_3 + 2x_1x_3^{10}\partial_2^3\partial_3 + 3x_1^2x_2^2x_3^7\partial_2\partial_3^3 - 2x_1^2x_2x_3^7\partial_1\partial_2\partial_3^3 +$
$3x_1x_2^2x_3^7\partial_1\partial_2\partial_3^3 - x_1^2x_3^8\partial_2^2\partial_3^3 + 3x_1x_3^8\partial_1\partial_2^2\partial_3^3 + 2x_2x_3^8\partial_1\partial_2^2\partial_3^3 + 2x_1^2x_3^7\partial_2^3\partial_3^3 - 3x_1x_3^8\partial_2^3\partial_3^3 + 2x_1x_3^7\partial_1\partial_2^3\partial_3^3 -$
$2x_3^8\partial_1\partial_2^3\partial_3^3 + 3x_1^2x_2^3x_3^9 + x_1x_2^3x_3^9\partial_1 + 3x_1^2x_2x_3^9\partial_1\partial_2 + 2x_1^2x_3^{10}\partial_2^2 + x_1x_3^{10}\partial_1\partial_2^2 + 3x_2x_3^{10}\partial_1\partial_2^2 - x_1x_3^{10}\partial_2^3 -$
$3x_3^{10}\partial_1\partial_2^3 + 3x_1x_3^{10}\partial_2^2\partial_3 + x_2x_3^{10}\partial_2^2\partial_3 - x_3^{10}\partial_2^3\partial_3 - x_1^2x_2^2x_3^7\partial_3^3 - 2x_1^2x_2x_3^7\partial_1\partial_3^3 - x_1x_2^2x_3^7\partial_1\partial_3^3 +$
$2x_1^2x_2x_3^7\partial_2\partial_3^3 - x_1x_2x_3^7\partial_1\partial_2\partial_3^3 + 2x_1^2x_3^7\partial_2^2\partial_3^3 + 2x_1x_3^7\partial_1\partial_2^2\partial_3^3 - 3x_1x_2^3x_3^9 + 2x_1^2x_2x_3^9\partial_2 + 3x_1x_2x_3^9\partial_1\partial_2 +$
$2x_3^{10}\partial_2^2\partial_3 + 2x_1^2x_2x_3^7\partial_3^3 - x_1x_2x_3^7\partial_1\partial_3^3 + x_1^2x_3^7\partial_2\partial_3^3 - x_1x_2x_3^7\partial_2\partial_3^3 + x_1x_3^7\partial_1\partial_2\partial_3^3 + x_1x_3^8\partial_3^4 - 2x_2^3x_3^9 -$
$2x_1x_2x_3^9\partial_2 - 2x_1x_3^{10}\partial_3 - 2x_1^2x_3^7\partial_3^3 - x_1x_2x_3^7\partial_3^3 - 2x_1x_3^7\partial_1\partial_3^3 - 3x_2x_3^7\partial_2\partial_3^3 - x_3^8\partial_2\partial_3^3 - 3x_1x_3^7\partial_3^4 +$
$x_2x_3^9\partial_2 + 2x_3^{10}\partial_2 - x_1x_3^9\partial_3 - 2x_1x_3^7\partial_3^3 - 3x_2x_3^7\partial_3^3 - x_1x_2\partial_1^3\partial_2\partial_3^5 + 2x_1x_2^2x_3^3\partial_2\partial_3^3 - x_1x_3^3\partial_2^3\partial_3^3 -$
$x_1x_2\partial_1^3\partial_3^5 - 2x_1^2x_2\partial_2^2\partial_3^5 - 2x_1x_2\partial_1\partial_2^2\partial_3^5 + 2x_1^2\partial_2^3\partial_3^5 + 2x_1\partial_1\partial_2^3\partial_3^5 - 2x_1x_3^8 - 3x_1x_2^2x_3^3\partial_3^3 + 3x_1x_2x_3^3\partial_2\partial_3^3 -$

$x_1x_3^3\partial_2^2\partial_3^3 - 2x_1\partial_1^3\partial_3^5 - 3x_1^2\partial_2^2\partial_3^5 + 3x_1x_2\partial_2^2\partial_3^5 - x_1x_2x_3\partial_1^3\partial_2\partial_3 + 3x_1x_2x_3^3\partial_3^3 + 3x_1x_3^3\partial_2\partial_3^3 +$
$x_2^2x_3^4\partial_2 + 3x_3^4\partial_2^3 + x_1^3x_2x_3\partial_1\partial_3 - x_1x_2x_3\partial_1^3\partial_3 - 2x_1^2x_2x_3\partial_2^2\partial_3 - 2x_1x_2x_3\partial_1\partial_2^2\partial_3 + 2x_1^2x_3\partial_2^3\partial_3 +$
$2x_1x_3\partial_1\partial_2^3\partial_3 + x_1x_3^3\partial_3^3 + 2x_2^2x_3^4 - 2x_2x_3^4\partial_2 + x_1^2\partial_1^3\partial_2 - x_1x_2\partial_1^3\partial_2 + 3x_3^4\partial_2^2 - 2x_1x_2^2\partial_2^3 - x_2^2\partial_2^4 +$
$x_1\partial_2^5 - 3\partial_2^6 + 3x_1^3x_2x_3\partial_3 + x_1^2x_2x_3\partial_1\partial_3 - 2x_1x_3\partial_1^3\partial_3 - 3x_1^2x_3\partial_2^2\partial_3 + 3x_1x_2x_3\partial_2^2\partial_3 + 3x_1^2x_2\partial_1\partial_3^2 -$
$3x_2^3\partial_2\partial_3^2 - 2x_2\partial_2^3\partial_3^2 + 3\partial_1^2\partial_3^4 - 3x_2\partial_2\partial_3^4 + x_1\partial_3^5 - 2x_2x_3^4 - x_1x_2\partial_1^3 - x_2^3x_3\partial_2 + x_2^2x_3^2\partial_2 - 2x_3^4\partial_2 +$
$3x_1^2\partial_1^2\partial_2 + x_1\partial_1^3\partial_2 - 2x_1^2x_2\partial_2^2 + 3x_1x_2^2\partial_2^2 - 2x_1x_2\partial_1\partial_2^2 + 2x_1^2\partial_2^3 - 3x_1x_2\partial_2^3 - 2x_2^2\partial_2^3 - 3x_2x_3\partial_2^3 +$
$3x_3^2\partial_2^3 + 2x_1\partial_1\partial_2^3 + x_1\partial_2^4 + 2x_2\partial_2^4 - 3\partial_2^5 - 3x_1^2x_2x_3\partial_3 + 2x_1^3\partial_1\partial_3 + 2x_1^2x_2\partial_3^2 + x_2^3\partial_3^2 + 3x_1x_2\partial_1\partial_3^2 -$
$x_2^2\partial_2\partial_3^2 - 2x_2\partial_2^2\partial_3^2 - 3x_3\partial_3^4 - 2x_2^3x_3 + 2x_2^2x_3^2 - 3x_3^4 - 3x_1^2x_2\partial_1 - 2x_1\partial_1^3 + 2x_2^2x_3\partial_2 - 2x_2x_3^2\partial_2 +$
$3x_2^2\partial_1\partial_2 + x_1\partial_1^2\partial_2 - 3x_1^2\partial_2^2 - x_1x_2\partial_2^2 - 3x_2x_3\partial_2^2 + 3x_3^2\partial_2^2 - 3x_1\partial_2^3 + 3x_2\partial_2^3 + 2\partial_1\partial_2^3 + 2\partial_2^4 -$
$x_1^3\partial_3 - 2x_1x_2x_3\partial_3 - 2x_1x_2\partial_3^2 - x_2^2\partial_3^2 - x_2\partial_2\partial_3^2 - 2x_1^2x_2 + 2x_2^2x_3 - 2x_2x_3^2 - 3x_1x_2\partial_1 - x_2^2\partial_1 +$
$3x_3\partial_1^2 - 2x_1x_2\partial_2 - x_2x_3\partial_2 - 2x_3^2\partial_2 - 2x_1\partial_1\partial_2 + x_2\partial_1\partial_2 + 2x_2\partial_2^2 + 2\partial_1\partial_2^2 + 2\partial_2^3 + 2x_1^2\partial_3 + x_1x_3\partial_3 -$
$2x_1\partial_1\partial_3 + 3x_2\partial_3^2 + 2x_1x_2 + 3x_2x_3 + x_3^2 + x_2\partial_1 - 3x_1\partial_2 - 3\partial_1\partial_2 + 2x_1\partial_3 - x_2 - 2\partial_1 + \partial_2 - 3\partial_3$

C.3 Chapter 6

(1) **Example 6.2.2**

The generating set $\{p_1, p_2\}$ consists of the following Weyl polynomials in A_3

$p_1 = x_1^6x_2^5x_3\partial_1^6 + x_1^6x_2^4x_3\partial_1^7 + x_1^5x_2^5\partial_1^7\partial_2 - x_1^3x_2^6x_3^3\partial_1^2\partial_2^4 - x_1^7x_2\partial_1^6\partial_2^4 + x_1^3x_2^8\partial_1\partial_2^6 + x_2^3x_3^6\partial_1^2\partial_2^7 -$
$x_2^5x_3^3\partial_1\partial_2^9 + x_1^8x_3^3\partial_1^6\partial_3 + x_1^5x_2^2\partial_1^7\partial_2^3\partial_3 + x_1^{10}\partial_1^6\partial_3^2 + x_1^7x_2^3\partial_1^6\partial_3^2 + x_1^3x_2^3x_3^6\partial_1^2\partial_2\partial_3^3 - x_1^3x_2^5x_3^3\partial_1\partial_2^3\partial_3^3 -$
$x_1^5x_2x_3^2\partial_1^9 + x_1^6x_2^2x_3\partial_1^6\partial_2^2 - x_1^5x_2^9\partial_2^3 + x_1^3x_2x_3\partial_1^7\partial_2^5 + x_1^2x_2^6x_3^3\partial_2^6 - x_1^4x_2^2\partial_1^9\partial_2\partial_3 + x_1^6x_2\partial_1^6\partial_2^3\partial_3 +$
$x_1^3x_3^6\partial_1^6\partial_3^2 - x_1^4x_2^3x_3\partial_1^7\partial_3^2 + x_1^3x_3^2\partial_1^{10}\partial_3^2 + x_1^5x_2^6x_3^3\partial_3^3 + x_1^5x_3^3\partial_1^6\partial_3^3 - x_1^3x_2x_3^2\partial_1^7\partial_2\partial_3^3 - x_1^3x_3^5\partial_1^6\partial_2\partial_3 -$
$x_1^3x_3\partial_1^8\partial_3^4 - x_1^6x_2^3x_3^3\partial_1^2\partial_2 + x_1^6x_2^5\partial_1\partial_2^3 - x_1^3x_2^6x_3\partial_1^2\partial_2^3 - x_1^2x_2^8\partial_1\partial_2^4 + x_1^3x_2^3x_3^3\partial_1^2\partial_2^4 - x_2^5x_3^4\partial_2^6 - x_1^3x_2^5\partial_1\partial_2^6 -$
$x_2^4x_3^4\partial_1\partial_2^6 + x_2^3x_3^4\partial_1^2\partial_2^6 + x_1^4x_2^4\partial_2^7 - x_1^5x_2^3x_3^3\partial_2^3\partial_3 - x_1^3x_2\partial_1^7\partial_2^3\partial_3 - x_1^3\partial_1^6\partial_2^5\partial_3 - x_1^2x_2^5\partial_1\partial_2^6\partial_3 - x_1^3x_2x_3\partial_1^6\partial_2^2\partial_3^2 -$
$x_1^7x_2^3\partial_2^3\partial_3^2 - x_1x_2^3x_3^3\partial_2^6\partial_3^2 + x_1^3x_2^3x_3^4\partial_1^2\partial_3^3 + x_1^4x_3^2\partial_1^6\partial_3^3 + x_1^2x_2^5x_3^3\partial_1\partial_2\partial_3^3 - x_1^4x_2x_3^3\partial_2^4\partial_3^3 + x_1^5x_3^6\partial_3^4 +$
$x_1^2x_2^2x_3^3\partial_1\partial_2^3\partial_3^4 + x_1^7x_3^3\partial_3^5 - x_3^6\partial_1^2\partial_2\partial_3^6 + x_2^2x_3^3\partial_1\partial_2^3\partial_3^6 - x_1^8x_2^6 + x_1^3x_3^3\partial_1^6\partial_2^2 + x_1^2x_2^4x_3^2\partial_1^3\partial_2^3 - x_1^3x_2^5x_3\partial_2^5 +$
$x_1^2x_2^3x_3^3\partial_2^6 - x_2^4x_3\partial_1\partial_2^8 + x_1x_2^5\partial_1^3\partial_2^4\partial_3 - x_1^3x_2^4\partial_2^6\partial_3 - x_2^3x_3^6\partial_2^3\partial_3^2 + x_1x_2^6x_3\partial_1\partial_2^3\partial_3^2 - x_2^3x_3^2\partial_1^4\partial_2^3\partial_3^2 +$
$x_1^5x_2^3x_3^3\partial_3^3 - x_1^2x_2x_3^5\partial_1^3\partial_3^3 + x_1^3x_2^2x_3^4\partial_2^2\partial_3^3 - x_1^2x_2^3x_3^3\partial_2^3\partial_3^3 + x_2^4x_3^2\partial_1\partial_2^4\partial_3^3 + x_2x_3^4\partial_1\partial_2^5\partial_3^3 - x_1x_2^2x_3^3\partial_1^3\partial_2\partial_3^4 +$
$x_1^3x_2x_3^3\partial_2^3\partial_3^4 + x_3^9\partial_3^5 - x_1x_2^3x_3^4\partial_1\partial_3^5 + x_3^5\partial_1^4\partial_3^5 - x_1^2x_2^3x_3^3\partial_3^6 + x_1^2x_3^6\partial_3^6 - x_2x_3^5\partial_1\partial_2\partial_3^6 + x_1^3x_2x_3^2\partial_1^7 +$
$x_2^3x_3^5\partial_2^4\partial_3 - x_3^8\partial_2\partial_3^4 + x_2^3x_3\partial_1^2\partial_2^3\partial_3^4 - x_3^4\partial_1^2\partial_3^7 - x_1^6x_2^3x_3\partial_1^2 - x_1^5x_2^5\partial_1\partial_2 - x_1^3x_2^3x_3^3\partial_1^2\partial_2 + x_1^3x_2^5\partial_1\partial_2^3 +$
$x_1^3x_2^3x_3\partial_1^2\partial_2^3 + x_1^7x_2\partial_2^4 + x_1^2x_2^5\partial_1\partial_2^4 - x_1^4x_2\partial_2^7 - x_1^8x_3^3\partial_3 + x_1^5x_3^3\partial_2^3\partial_3 - x_1^5x_2^2\partial_1\partial_2^3\partial_3 + x_1^2x_2^2\partial_1\partial_2^6\partial_3 +$
$x_2^4\partial_1\partial_2^6\partial_3 + x_2^3\partial_2^8\partial_3 - x_1^{10}\partial_3^2 + x_1^7\partial_2^3\partial_3^2 + x_2^4x_3\partial_2^5\partial_3^2 - x_1x_2^3x_3^2\partial_2^3\partial_3^3 - x_2x_3^3\partial_1\partial_2^3\partial_3^4 - x_3^3\partial_2^5\partial_3^4 -$
$x_2x_3^4\partial_2^2\partial_3^5 + x_2^2x_3^4\partial_3^6 + x_1x_3^5\partial_3^6 + x_2x_3^4\partial_1\partial_3^6 - x_3^4\partial_1^2\partial_3^6 + x_1x_3^3\partial_3^8 - x_1^8x_2^3 - x_1^5x_2^6 + x_1^5x_2x_3^2\partial_1^3 - x_1^6x_2^2x_3\partial_2^2 +$

$x_1^5x_2^3\partial_2^3 - x_1^2x_2x_3^2\partial_1^3\partial_2^3 + x_1^3x_2^2x_3\partial_2^5 - x_2^3x_3^3\partial_2^5 - x_1^3x_2x_3\partial_1\partial_2^5 + x_2x_3\partial_1\partial_2^8 + x_1^4x_2^2\partial_1^3\partial_2\partial_3 - x_1^6x_2\partial_2^3\partial_3 -$
$x_1x_2^2\partial_1^3\partial_2^4\partial_3 + x_1^3x_2\partial_2^6\partial_3 - x_1^3x_3^6\partial_3^2 + x_1^4x_2^3x_3\partial_1\partial_3^2 - x_1^3x_3^2\partial_1^4\partial_3^2 + x_3^6\partial_2^3\partial_3^2 - x_1x_2^3x_3\partial_1\partial_2^3\partial_3^2 + x_3^2\partial_1^4\partial_2^3\partial_3^2 -$
$x_1^5x_3^3\partial_3^3 + x_1^3x_2x_3^2\partial_1\partial_2\partial_3^3 + x_3^6\partial_2^2\partial_3^3 + x_1^2x_3^3\partial_2^3\partial_3^3 - x_2x_3^2\partial_1\partial_2^4\partial_3^3 - x_1^2x_3^3\partial_3^6 - x_2^4x_3^2\partial_1\partial_2^3 + x_1^3x_3^5\partial_2\partial_3 -$
$x_3^5\partial_2^4\partial_3 + x_2x_3^5\partial_1\partial_3^3 + x_1^3x_3\partial_1^2\partial_3^4 - x_3\partial_1^2\partial_2^3\partial_3^4 - x_1^3x_2^3x_3\partial_1^2 - x_1^2x_2^5\partial_1\partial_2 + x_3^3\partial_1^5\partial_2 - x_2^2\partial_1^4\partial_2^3 + x_1^4x_2\partial_2^4 -$
$x_1^5x_3^3\partial_3 + x_1^3x_2\partial_1\partial_2^3\partial_3 - x_1^2x_2^2\partial_1\partial_2^3\partial_3 + x_1^3\partial_2^5\partial_3 - x_2\partial_1\partial_2^6\partial_3 - \partial_2^8\partial_3 - x_1^7\partial_3^2 + x_1^3x_2x_3\partial_2^2\partial_3^2 - x_2x_3\partial_2^5\partial_3^2 -$
$x_1^4x_3^2\partial_3^3 - x_3^3\partial_1^2\partial_2\partial_3^3 + x_1x_3^2\partial_2^3\partial_3^3 + x_2^2\partial_1\partial_2^3\partial_3^3 - x_1^5x_2^3 + x_1^2x_2^3\partial_1^3 + x_1^2x_2x_3^2\partial_1^3 - x_1^3x_2^2x_3\partial_2^2 - x_1^3x_3^3\partial_2^2 +$
$x_3^3\partial_2^5 - x_2x_3\partial_1\partial_2^5 + x_1x_2^2\partial_1^3\partial_2\partial_3 - x_1^3x_2\partial_2^3\partial_3 - x_3^6\partial_3^2 + x_1x_2^3x_3\partial_1\partial_3^2 - x_3^2\partial_1^4\partial_3^2 - x_1^2x_2^3\partial_3^3 - x_1^2x_3^3\partial_3^3 +$
$x_2x_3^2\partial_1\partial_2\partial_3^3 - x_1^3x_2x_3^2\partial_1 + x_2x_3^2\partial_1\partial_2^3 + x_3^5\partial_2\partial_3 + x_3\partial_1^2\partial_3^4 - x_2^2x_3\partial_1^3 - x_2x_3\partial_1^4 + x_3\partial_1^5 + x_3^3\partial_1^2\partial_2 -$
$x_2^2\partial_1\partial_2^3 + x_2\partial_1\partial_2^3\partial_3 + \partial_2^5\partial_3 - x_1\partial_1^3\partial_3^2 + x_2x_3\partial_2^2\partial_3^2 + x_2^2x_3\partial_3^3 - x_1x_3^2\partial_3^3 + x_2x_3\partial_1\partial_3^3 - x_3\partial_1^2\partial_3^3 +$
$x_1\partial_3^5 + x_1^2x_2^3 + x_1^2\partial_1^3 - x_3^3\partial_2^2 - x_1^2\partial_3^3 - x_2x_3^2\partial_1 - x_2^2x_3 - x_2x_3\partial_1 + x_3\partial_1^2 - x_1\partial_3^2 + x_1^2,$

$p_2 = x_1^5x_2x_3^4\partial_1^7 - x_1^3x_2^8x_3\partial_1^2\partial_2^3 + x_1^4x_2x_3^2\partial_1^7\partial_2^3 + x_1^6x_2^6\partial_1\partial_2^4 + x_2^5x_3^4\partial_1^2\partial_2^6 - x_1^3x_2^3x_3^3\partial_1\partial_2^7 - x_1^6\partial_1^6\partial_2^4\partial_3 -$
$x_1^3x_2^4\partial_1^8\partial_3^2 + x_1^5x_2^3\partial_1^6\partial_2\partial_3^2 + x_1^5x_2^2x_3\partial_1^6\partial_2\partial_3^2 + x_1^5x_2\partial_1^6\partial_2^3\partial_3^2 + x_1^3x_2^5x_3^4\partial_1^2\partial_3^3 - x_1^6x_2^3x_3^3\partial_1\partial_2\partial_3^3 - x_1^3x_2^2x_3\partial_1^7\partial_2\partial_3^3 +$
$x_1^3x_2\partial_1^6\partial_3^7 + x_1^3x_2^6\partial_1^7 - x_1^4x_2^4\partial_1^7\partial_3 + x_1^4x_2^3x_3\partial_1^7\partial_3 + x_1^3x_2\partial_1^6\partial_2^5\partial_3 - x_1^6x_3\partial_1^6\partial_2\partial_3^2 - x_1^4x_2x_3\partial_1^6\partial_2^2\partial_3^2 -$
$x_1^4x_2^2x_3\partial_1^6\partial_3^3 + x_1^4\partial_1^6\partial_2^3\partial_3^3 + x_1^7x_3\partial_1^7 - x_1^3x_2^3\partial_1^6\partial_2^3 - x_1^4x_2^2\partial_1^6\partial_2^2\partial_3 - x_1^3x_2^7\partial_1\partial_2^3\partial_3 + x_2^4x_3^3\partial_1\partial_2^6\partial_3 +$
$x_1^5x_3^2\partial_1^6\partial_3^2 + x_1^3x_2\partial_1^7\partial_2\partial_3^3 + x_1^3x_2^4x_3^3\partial_1\partial_3^4 - x_1^3x_2x_3\partial_1^6\partial_3^4 - x_1^6x_2^5x_3\partial_1^2 + x_1^9x_2^3\partial_1\partial_2 - x_1^2x_2^4x_3^4\partial_1\partial_2^3 +$
$x_1^3x_2^5x_3\partial_1^2\partial_2^3 - x_1^6x_2^3\partial_1\partial_2^4 - x_1x_2^4x_3^2\partial_1\partial_2^6 + x_1^6x_3\partial_1^6\partial_3 + x_1^4x_2x_3\partial_1^6\partial_2\partial_3 + x_1^3x_2x_3^2\partial_1^6\partial_2\partial_3 + x_1^3x_2^3\partial_2^7\partial_3 -$
$x_1^5\partial_1^7\partial_3^2 + x_2^7\partial_1^2\partial_2^3\partial_3^2 - x_1^2x_2^6\partial_2^4\partial_3^2 - x_1^2x_2^5x_3\partial_2^4\partial_3^2 - x_1^2x_2^4\partial_2^6\partial_3^2 + x_1^2x_2x_3^7\partial_1\partial_3^3 + x_1x_2x_3^5\partial_1\partial_2^3\partial_3^3 +$
$x_2^5x_3\partial_1\partial_2^4\partial_3^3 - x_1^3x_3^3\partial_2^4\partial_3^4 - x_2^4x_3^3\partial_1^2\partial_3^5 + x_1^2x_2^3x_3^3\partial_2\partial_3^5 + x_1^2x_2^2x_3^4\partial_2\partial_3^5 + x_1^2x_2x_3^3\partial_2^3\partial_3^5 - x_2^2x_3^4\partial_1^2\partial_3^6 +$
$x_1^3x_3^3\partial_1\partial_2\partial_3^6 - x_2^2x_3^4\partial_1\partial_2\partial_3^6 - x_2^4\partial_2^3\partial_3^7 + x_2x_3^3\partial_3^{10} - x_2^9\partial_1\partial_2^3 + x_1x_2^7\partial_1\partial_2^3\partial_3 - x_1x_2^6x_3\partial_1\partial_2^3\partial_3 - x_2^4\partial_2^8\partial_3 +$
$x_1^3x_2^3x_3\partial_2^4\partial_3^2 + x_1x_2^4x_3\partial_2^5\partial_3^2 + x_2^6x_3^3\partial_1\partial_3^3 - x_1^3x_3\partial_1^6\partial_3^3 + x_1x_2^5x_3\partial_2^3\partial_3^3 - x_1x_2^3\partial_2^6\partial_3^3 - x_1x_2^4x_3^3\partial_1\partial_3^4 +$
$x_1x_2^3x_3^4\partial_1\partial_3^4 + x_2x_3^3\partial_2^5\partial_3^4 - x_1^3x_3^4\partial_2\partial_3^5 - x_1x_2x_3^4\partial_2^2\partial_3^5 - x_1x_2^2x_3^4\partial_3^6 + x_1x_3^3\partial_2^3\partial_3^6 - x_1^4x_2^3x_3\partial_1\partial_2^3 +$
$x_2^6\partial_2^6 - x_1^6x_2^4\partial_1\partial_3 + x_1^3x_2^4\partial_1\partial_2^3\partial_3 + x_1x_2^5\partial_2^5\partial_3 - x_1^2x_2^3x_3^2\partial_2^3\partial_3^2 + x_1^4x_3^4\partial_1\partial_3^3 - x_2^3x_3^3\partial_2^3\partial_3^3 - x_2^4\partial_1\partial_2^4\partial_3^3 -$
$x_1x_2^2x_3^3\partial_2^2\partial_3^4 + x_2^4x_3\partial_2^3\partial_3^4 + x_1^2x_3^5\partial_3^5 + x_2x_3^3\partial_1\partial_2\partial_3^6 - x_2x_3^4\partial_3^7 - x_2x_3^3\partial_1\partial_3^7 - x_1^5x_2x_3^4\partial_1 - x_1^3x_2^5x_3\partial_1^2 +$
$x_1^6x_2^3\partial_1\partial_2 - x_1^4x_2x_3^2\partial_1\partial_2^3 + x_1^2x_2x_3^4\partial_1\partial_2^3 + x_1x_2x_3^2\partial_1\partial_2^6 - x_1^3x_2^3x_3\partial_2^3\partial_3 + x_1^6\partial_2^4\partial_3 - x_1x_2^4x_3\partial_2^4\partial_3 -$
$x_2^4x_3^2\partial_2^4\partial_3 - x_1^3\partial_2^7\partial_3 + x_1^3x_2^4\partial_1^2\partial_3^2 - x_1^5x_2^3\partial_2\partial_3^2 - x_1^5x_2^2x_3\partial_2\partial_3^2 - x_1^5x_2\partial_2^3\partial_3^2 + x_1^2x_2^3\partial_1\partial_2^3\partial_3^2 - x_2^4\partial_1^2\partial_2^3\partial_3^2 +$
$x_1^2x_2^3\partial_2^4\partial_3^2 + x_1^2x_2^2x_3\partial_2^4\partial_3^2 + x_1^2x_2\partial_2^6\partial_3^2 + x_1^3x_2^2x_3\partial_1\partial_2\partial_3^3 - x_2^2x_3\partial_1\partial_2^4\partial_3^3 + x_1^3x_3^4\partial_3^4 + x_1x_2x_3^4\partial_2\partial_3^4 +$
$x_2x_3^5\partial_2\partial_3^4 - x_1^2x_3^3\partial_1\partial_3^5 - x_1^3x_2\partial_3^7 + x_2\partial_2^3\partial_3^7 - x_1^3x_2^6\partial_1 + x_2^6\partial_1\partial_2^3 + x_1^4x_2^4\partial_1\partial_3 - x_1^4x_2^3x_3\partial_1\partial_3 - x_1x_2^4\partial_1\partial_2^3\partial_3 +$
$x_1x_2^3x_3\partial_1\partial_2^3\partial_3 - x_1^3x_2\partial_2^5\partial_3 + x_2\partial_2^8\partial_3 + x_1^6x_3\partial_2\partial_3^2 + x_1^4x_2x_3\partial_2^2\partial_3^2 - x_1^3x_3\partial_2^4\partial_3^2 - x_1x_2x_3\partial_2^5\partial_3^2 +$
$x_1^4x_2^2x_3\partial_3^3 - x_1^4\partial_2^3\partial_3^3 - x_1x_2^2x_3\partial_2^3\partial_3^3 + x_2^3x_3\partial_2^3\partial_3^3 + x_1\partial_2^6\partial_3^3 - x_3^4\partial_3^6 - x_1^7x_3\partial_1 + x_1^3x_2^3\partial_2^3 + x_1^4x_3\partial_1\partial_2^3 -$
$x_2^3\partial_2^6 - x_1^3x_2^4\partial_1\partial_3 + x_1^4x_2^2\partial_2^2\partial_3 - x_1x_2^2\partial_2^5\partial_3 - x_1^5x_3^3\partial_3^2 + x_1^2x_3^2\partial_2^3\partial_3^2 - x_1^3x_2\partial_1\partial_2\partial_3^3 + x_2\partial_1\partial_2^4\partial_3^3 +$
$x_1^3x_2x_3\partial_3^4 - x_2x_3\partial_2^3\partial_3^4 - x_1^2x_2x_3^4\partial_1 + x_2^2x_3\partial_1^5 - x_1^3\partial_1^4\partial_2 - x_1x_2x_3^2\partial_1\partial_2^3 - x_1^6x_3\partial_3 - x_1^4x_2x_3\partial_2\partial_3 -$
$x_1^3x_2x_3^2\partial_2\partial_3 + x_1^3x_3\partial_2^3\partial_3 + x_1^3\partial_2^4\partial_3 + x_1x_2x_3\partial_2^4\partial_3 + x_2x_3^2\partial_2^4\partial_3 + x_1^5\partial_1\partial_3^2 + x_2^4\partial_1^2\partial_3^2 - x_1^2x_2^3\partial_2\partial_3^2 -$

$x_1^2x_2^2x_3\partial_2\partial_3^2 - x_1^2x_2\partial_2^3\partial_3^2 - x_1^2\partial_1\partial_2^3\partial_3^2 - x_2^2x_3\partial_1^2\partial_3^3 + x_1^3\partial_1\partial_2\partial_3^3 + x_2^2x_3\partial_1\partial_2\partial_3^3 - x_2\partial_3^7 - x_2^6\partial_1 +$
$x_1x_2^4\partial_1\partial_3 - x_1x_2^3x_3\partial_1\partial_3 - x_2\partial_2^5\partial_3 + x_1^3x_3\partial_2\partial_3^2 + x_1x_2x_3\partial_2^2\partial_3^2 + x_1^3x_3\partial_3^3 + x_1x_2^2x_3\partial_3^3 - x_1\partial_2^3\partial_3^3 -$
$x_3\partial_2^3\partial_3^3 - x_1^4x_3\partial_1 + x_2^3\partial_2^3 + x_2\partial_1^4\partial_3 + x_1x_2^2\partial_2^2\partial_3 - x_1^2x_3^2\partial_3^2 - x_2\partial_1\partial_2\partial_3^3 + x_2x_3\partial_3^4 - x_2\partial_1\partial_3^4 + x_2^2x_3\partial_1^2 -$
$x_1^3\partial_1\partial_2 - x_1^3x_3\partial_3 - x_1x_2x_3\partial_2\partial_3 - x_2x_3^2\partial_2\partial_3 + x_1^2\partial_1\partial_3^2 + x_3\partial_3^3 + x_2\partial_1\partial_3$

(2) **Example 6.3.2**

The polynomials p_1, and p_2 of the public key Q are:

$p_1 = 6x_1^{14}x_2^{25}\partial_1^{28}\partial_2 + x_1^{14}x_2^{24}\partial_1^{28} - 3x_1^{17}x_2^{16}\partial_1^{28}\partial_2^2 - 6x_1^{16}x_2^{19}\partial_1^{26} - 5x_1^{16}x_2^{16}\partial_1^{27}\partial_2^2 + x_1^{14}x_2^{16}\partial_1^{28}\partial_2^2 -$
$5x_1^{15}x_2^{16}\partial_1^{26}\partial_2^2 + 2x_1^{13}x_2^{19}\partial_1^{26} - 2x_1^{14}x_2^{10}\partial_1^{28}\partial_2^5 - 3x_1^{14}x_2^{10}\partial_1^{29}\partial_2^2 - 4x_1^{13}x_2^{13}\partial_1^{26}\partial_2^3 + 4x_1^{14}x_2^9\partial_1^{28}\partial_2^4 +$
$x_1^{14}x_2^{10}\partial_1^{28}\partial_2^2 - 6x_1^{13}x_2^{13}\partial_1^{27} + 5x_1^{14}x_2^9\partial_1^{29}\partial_2 - 3x_1^{13}x_2^{10}\partial_1^{28}\partial_2^2 - 6x_1^{14}x_2^8\partial_1^{28}\partial_2^3 + 2x_1^{13}x_2^{13}\partial_1^{26} - 6x_1^{14}x_2^9\partial_1^{28}\partial_2 +$
$3x_1^{14}x_2^8\partial_1^{29} + 5x_1^{13}x_2^9\partial_1^{28}\partial_2 + 4x_1^{14}x_2^7\partial_1^{28}\partial_2^2 - x_1^{14}x_2^8\partial_1^{28} - 6x_1^{17}\partial_1^{28}\partial_2^5 + 3x_1^{13}x_2^8\partial_1^{28} + x_1^{14}x_2^6\partial_1^{28}\partial_2 +$
$4x_1^{15}x_2^2\partial_1^{30}\partial_2^2 + 4x_1^{17}\partial_1^{29}\partial_2^2 + x_1^{16}x_2^3\partial_1^{26}\partial_2^3 + 3x_1^{16}\partial_1^{27}\partial_2^5 + 4x_1^{14}x_2^5\partial_1^{28} + 3x_1^{15}x_2\partial_1^{30}\partial_2 + 3x_1^{17}\partial_1^{28}\partial_2^2 +$
$4x_1^{14}x_2^2\partial_1^{29}\partial_2^2 + 2x_1^{14}\partial_1^{28}\partial_2^5 - 5x_1^{16}x_2^3\partial_1^{27} - 4x_1^{16}x_2^2\partial_1^{26}\partial_2^2 + x_1^{16}\partial_1^{28}\partial_2^2 + 3x_1^{15}\partial_1^{26}\partial_2^5 + 6x_1^{16}x_2^3\partial_1^{26} -$
$5x_1^{15}\partial_1^{30} + 3x_1^{14}x_2\partial_1^{29}\partial_2 + 5x_1^{16}\partial_1^{27}\partial_2^2 + 3x_1^{14}\partial_1^{29}\partial_2^2 + 4x_1^{13}x_2^3\partial_1^{26}\partial_2^3 - 2x_1^{15}x_2^3\partial_1^{26} + 5x_1^{16}x_2\partial_1^{26}\partial_2 +$
$5x_1^{15}\partial_1^{27}\partial_2^2 - x_1^{14}\partial_1^{28}\partial_2^2 + 6x_1^{13}x_2^3\partial_1^{27} - 5x_1^{14}\partial_1^{29} + 5x_1^{15}\partial_1^{26}\partial_2^2 - 3x_1^{13}x_2^2\partial_1^{26}\partial_2^2 + 3x_1^{13}\partial_1^{28}\partial_2^2 + 6x_1^{16}\partial_1^{26} -$
$2x_1^{13}x_2^3\partial_1^{26} - 4x_1^{14}\partial_1^{26}\partial_2^2 - x_2^{39}\partial_1^2 + 3x_2^{39}\partial_1\partial_2 - 6x_1^{13}x_2\partial_1^{26}\partial_2 + 3x_2^{40} + 3x_2^{39}\partial_2 + 5x_2^{39} + 4x_2^{38}\partial_1 -$
$2x_1^{13}\partial_1^{26} + 4x_2^{38} - 3x_1^3x_2^{29}\partial_1^2 - 4x_1^3x_2^{29}\partial_1\partial_2 - 4x_1^3x_2^{30} - 4x_1^3x_2^{29}\partial_2 + 2x_1^3x_2^{29} - 5x_1^2x_2^{29}\partial_1 + x_1^2x_2^{29}\partial_2 +$
$x_2^{29}\partial_1^2 - 3x_2^{29}\partial_1\partial_2 - 2x_1x_2^{26}\partial_1^2\partial_2^2 - 2x_1x_2^{13}\partial_1^2\partial_2^{15} - 5x_1x_2^{29} - 3x_2^{30} - 3x_2^{29}\partial_2 + 4x_2^{29} - x_1x_2^{25}\partial_1^2\partial_2 -$
$4x_2^{16}\partial_2^{13} - 5x_2^{26}\partial_1^2 + 2x_2^{26}\partial_1\partial_2 - 2x_2^{23}\partial_1^2\partial_2^3 + 6x_2^{23}\partial_1\partial_2^4 + 2x_2^{27} + 2x_1x_2^{24}\partial_1^2 + 2x_2^{26}\partial_2 + 6x_2^{24}\partial_2^3 +$
$6x_2^{23}\partial_2^4 - x_2^{26} - 6x_2^{25}\partial_1 - 3x_2^{23}\partial_1^3 - 4x_2^{23}\partial_1^2\partial_2 + 5x_2^{22}\partial_1^2\partial_2^2 - 3x_2^{23}\partial_2^3 + 6x_2^{22}\partial_1\partial_2^3 - 6x_2^{25} - 4x_2^{24}\partial_1 +$
$x_2^{23}\partial_1^2 + 6x_2^{23}\partial_1\partial_2 + 3x_2^{23}\partial_2^2 + 6x_2^{22}\partial_2^3 - 3x_1^4x_2^{15}\partial_1^2\partial_2^4 - 3x_2^{24} + 2x_2^{23}\partial_1 - x_2^{22}\partial_1^2 - 3x_2^{23}\partial_2 + 6x_2^{21}\partial_1^2\partial_2 +$
$x_2^{22}\partial_2^2 + 3x_2^{21}\partial_1\partial_2^2 - 6x_1^4x_2^{16}\partial_1^2\partial_2^2 - 5x_2^{23} - 5x_2^{22}\partial_1 + 4x_2^{22}\partial_2 - 6x_1^3x_2^{18}\partial_2^2 + 3x_2^{21}\partial_2^2 + x_1^4x_2^{14}\partial_1^2\partial_2^3 +$
$4x_1^3x_2^{15}\partial_1\partial_2^4 + x_1^3x_2^{19} - 4x_2^{22} + 3x_2^{20}\partial_1^2 - 4x_2^{21}\partial_2 + 3x_2^{20}\partial_1\partial_2 + 3x_1^3x_2^{16}\partial_1\partial_2^2 - x_2^{21} - 2x_1^3x_2^{16}\partial_1^2 +$
$3x_2^{20}\partial_2 + 6x_1^3x_2^{16}\partial_1\partial_2 - 6x_1^4x_2^{13}\partial_1^2\partial_2^2 + 2x_1x_2^{16}\partial_1^2\partial_2^2 + 3x_1^3x_2^{14}\partial_1\partial_2^3 - 6x_1^3x_2^{13}\partial_1^2\partial_2^3 + 5x_1^3x_2^{13}\partial_1\partial_2^4 +$
$x_1x_2^{13}\partial_1^3\partial_2^4 + 6x_1^3x_2^{17} - 2x_2^{20} + 2x_2^{19}\partial_1 + 6x_1^3x_2^{16}\partial_2 + 3x_1^2x_2^{16}\partial_2^2 - 3x_1x_2^{13}\partial_1^4\partial_2^2 + 5x_1^3x_2^{14}\partial_2^3 + 5x_1^3x_2^{13}\partial_2^4 +$
$2x_1x_2^{13}\partial_1^2\partial_2^4 - 3x_1^3x_2^{16} + 6x_2^{19} + x_1^2x_2^{16}\partial_1 + 4x_1^3x_2^{13}\partial_1^3 + 5x_1^2x_2^{16}\partial_2 + x_1^3x_2^{13}\partial_1^2\partial_2 - 5x_1^3x_2^{13}\partial_1\partial_2^2 +$
$2x_2^{16}\partial_1\partial_2^2 + 4x_1^3x_2^{13}\partial_2^3 + 3x_1^2x_2^{13}\partial_1\partial_2^3 + 2x_1^2x_2^{13}\partial_2^4 + x_1^3x_2^{14}\partial_1 + 3x_1^3x_2^{13}\partial_1^2 - x_2^{16}\partial_1^2 + 5x_1^3x_2^{13}\partial_1\partial_2 -$
$2x_2^{16}\partial_1\partial_2 + 2x_1^3x_2^{13}\partial_2^2 + 4x_2^{16}\partial_2^2 + 2x_2^{13}\partial_1^2\partial_2^3 - 6x_2^{13}\partial_1\partial_2^4 - 4x_1x_2^{10}\partial_1^2\partial_2^5 + 3x_1\partial_1^2\partial_2^{15} + 4x_1^3x_2^{14} +$
$x_1x_2^{16} - 2x_2^{17} + 6x_1^3x_2^{13}\partial_1 - 3x_1^2x_2^{13}\partial_1^2 + 4x_1^3x_2^{13}\partial_2 - 2x_2^{16}\partial_2 + 6x_1^2x_2^{13}\partial_1\partial_2 + 3x_1x_2^{13}\partial_2^3 - 6x_2^{14}\partial_2^3 -$
$6x_2^{13}\partial_2^4 - 2x_1^3x_2^{13} + 3x_1^2x_2^{14} + x_2^{16} + 5x_1^2x_2^{13}\partial_1 + 3x_2^{13}\partial_1^3 + 2x_1^2x_2^{13}\partial_2 + 4x_2^{13}\partial_1^2\partial_2 - 6x_1x_2^{10}\partial_1^3\partial_2^2 -$
$5x_2^{13}\partial_2^3 - 5x_1x_2^9\partial_1^2\partial_2^4 + 6x_2^3\partial_2^{13} + 5x_1^2x_2^{13} - 6x_1x_2^{13}\partial_1 + 4x_2^{14}\partial_1 - x_2^{13}\partial_1^2 + 6x_1x_2^{13}\partial_2 - 6x_2^{13}\partial_1\partial_2 -$
$5x_2^{13}\partial_2^2 + 2x_1x_2^{10}\partial_1^2\partial_2^2 + 3x_2^{10}\partial_1^2\partial_2^3 + 4x_2^{10}\partial_1\partial_2^4 + 5x_1x_2^{13} + 3x_2^{14} - x_2^{13}\partial_1 + 3x_2^{13}\partial_2 - 3x_1x_2^9\partial_1^3\partial_2 -$
$6x_2^{10}\partial_1^2\partial_2^2 + 4x_2^{11}\partial_2^3 + x_1x_2^8\partial_1^2\partial_2^3 + 4x_2^{10}\partial_2^4 - 6x_2^{13} - 2x_2^{10}\partial_1^3 + x_1x_2^9\partial_1^2\partial_2 + 6x_2^{10}\partial_1^2\partial_2 - x_2^9\partial_1^2\partial_2^2 -$

$2x_2^{10}\partial_2^3+4x_2^9\partial_1\partial_2^3+6x_2^{11}\partial_1+5x_2^{10}\partial_1^2+6x_1x_2^8\partial_1^3+4x_2^{10}\partial_1\partial_2-3x_2^9\partial_1^2\partial_2+2x_2^{10}\partial_2^2-5x_1x_2^7\partial_1^2\partial_2^2+$
$4x_2^9\partial_2^3-2x_1^4x_2^2\partial_1^2\partial_2^4-2x_2^{11}-3x_2^{10}\partial_1-2x_1x_2^8\partial_1^2-5x_2^9\partial_1^2-2x_2^{10}\partial_2+4x_2^8\partial_1^2\partial_2+5x_2^9\partial_2^2+2x_2^8\partial_1\partial_2^2+$
$x_1^4\partial_1^2\partial_2^5+x_2^{10}+x_2^9\partial_1+6x_2^8\partial_1^2-6x_2^9\partial_2+2x_1x_2^6\partial_1^2\partial_2-4x_1^3x_2^5\partial_2^2+2x_2^8\partial_2^2-5x_1^2x_2^2\partial_1^4\partial_2^2+5x_1^4x_2\partial_1^2\partial_2^3-$
$6x_1^3x_2^2\partial_1\partial_2^4+6x_2^9+2x_2^7\partial_1^2+6x_2^8\partial_2+2x_2^7\partial_1\partial_2-5x_1^4\partial_1^3\partial_2^2+2x_1^3x_2^3\partial_2^3+6x_1^3\partial_1\partial_2^5-5x_2^8-5x_1x_2^5\partial_1^2+$
$2x_2^7\partial_2+6x_1^2x_2\partial_1^4\partial_2+2x_1^4\partial_1^2\partial_2^2-5x_1x_2^2\partial_1^3\partial_2^2+2x_1^3x_2\partial_1\partial_2^3-4x_1^3\partial_1^2\partial_2^3-x_1^3\partial_1\partial_2^4+5x_1\partial_1^3\partial_2^4+$
$4x_1\partial_1^2\partial_2^5+3x_2^7+3x_1^3x_2^3\partial_1-3x_2^6\partial_1+5x_1^3x_2^2\partial_2^2+2x_1^3\partial_1^2\partial_2^2-2x_1\partial_1^4\partial_2^2-x_1^3x_2\partial_2^3-x_1^3\partial_2^4-3x_1\partial_1^2\partial_2^4+$
$6x_1^2\partial_2^5-x_1^3x_2^3-3x_2^6-6x_1^3\partial_1^3+3x_1^2\partial_1^4+5x_1^3\partial_1^2\partial_2+6x_1x_2\partial_1^3\partial_2-2x_1^3\partial_1\partial_2^2-3x_2^3\partial_1\partial_2^2+6x_1\partial_1^3\partial_2^2-$
$6x_1^3\partial_2^3-5x_2^3\partial_2^3+2x_1^2\partial_1\partial_2^3-3x_1^2\partial_2^4-4x_1^2x_2^3+5x_1^3x_2\partial_1+2x_1^3\partial_1^2-4x_2^3\partial_1^2-3x_1^3x_2\partial_2-x_1^3\partial_1\partial_2-$
$3x_1^3\partial_2^2-6x_2^3\partial_2^2-3x_1^2\partial_1\partial_2^2-4x_1\partial_1^2\partial_2^2-3\partial_1^2\partial_2^3-4\partial_1\partial_2^4-6x_1^3x_2+4x_1^3\partial_1-x_2^3\partial_1-2x_1^2\partial_1^2+$
$3x_1\partial_1^3-6x_1^3\partial_2+4x_1^2\partial_1\partial_2-3x_1^2\partial_2^2-6x_2^2\partial_2^2+6\partial_1^2\partial_2^2+2x_1\partial_2^3-4x_2\partial_2^3-4\partial_2^4+2x_1^3+2x_1^2x_2+$
$5x_2^3-x_1^2\partial_1+2\partial_1^3-3x_1^2\partial_2-6\partial_1^2\partial_2+5x_1\partial_2^2+2\partial_2^3-x_1^2-4x_1\partial_1-6x_2\partial_1-5\partial_1^2+4x_1\partial_2+x_2\partial_2-$
$4\partial_1\partial_2+\partial_2^2-x_1+2x_2+3\partial_1+2\partial_2-2$, and

$p_2=-x_1^{14}x_2^{32}\partial_1^{28}\partial_2^3-2x_1^{13}x_2^{35}\partial_1^{26}\partial_2-x_1^{14}x_2^{31}\partial_1^{28}\partial_2^2-3x_1^{14}x_2^{30}\partial_1^{28}\partial_2+5x_1^{13}x_2^{28}\partial_1^{28}\partial_2^4-2x_1^{13}x_2^{31}\partial_1^{26}\partial_2^2-$
$4x_1^{14}x_2^{27}\partial_1^{28}\partial_2^3+3x_1^{16}x_2^{26}\partial_1^{27}\partial_2^2-2x_1^{14}x_2^{28}\partial_1^{27}\partial_2^2-6x_1^{13}x_2^{27}\partial_1^{28}\partial_2^3+6x_1^{13}x_2^{30}\partial_1^{26}\partial_2+2x_1^{15}x_2^{26}\partial_1^{27}\partial_2^2+$
$5x_1^{13}x_2^{28}\partial_1^{27}\partial_2^2+6x_1^{14}x_2^{26}\partial_1^{28}\partial_2^2+5x_1^{14}x_2^{27}\partial_1^{27}\partial_2+6x_1^{15}x_2^{26}\partial_1^{26}\partial_2^2+x_1^{14}x_2^{26}\partial_1^{27}\partial_2^2-3x_1^{13}x_2^{26}\partial_1^{28}\partial_2^2-$
$x_1^{13}x_2^{27}\partial_1^{27}\partial_2-2x_1^{14}x_2^{25}\partial_1^{28}\partial_2+4x_1^{14}x_2^{26}\partial_1^{26}\partial_2^2+5x_1^{13}x_2^{26}\partial_1^{27}\partial_2^2+6x_1^{15}x_2^{26}\partial_1^{26}-4x_1^{13}x_2^{28}\partial_1^{26}-4x_1^{14}x_2^{26}\partial_1^{27}-$
$x_1^{14}x_2^{23}\partial_1^{28}\partial_2^2+6x_1^{13}x_2^{26}\partial_1^{27}+4x_1^{14}x_2^{22}\partial_1^{28}\partial_2-x_1^{14}x_2^{19}\partial_1^{28}\partial_2^3-6x_1^{14}x_2^{21}\partial_1^{28}-2x_1^{13}x_2^{22}\partial_1^{26}\partial_2+6x_1^{14}x_2^{18}\partial_1^{28}\partial_2^2-$
$2x_1^{13}x_2^{21}\partial_1^{26}-5x_1^{14}x_2^{17}\partial_1^{28}\partial_2-2x_1^{14}x_2^{16}\partial_1^{28}+4x_1^{15}x_2^4\partial_1^{30}\partial_2^4-6x_1^{14}x_2^4\partial_1^{30}\partial_2^4+6x_1^{15}x_2^3\partial_1^{30}\partial_2^3+4x_1^{14}x_2^4\partial_1^{29}\partial_2^4-$
$x_1^{17}x_2^2\partial_1^{29}\partial_2^2+5x_1^{15}x_2^4\partial_1^{29}\partial_2^2+4x_1^{14}x_2^3\partial_1^{30}\partial_2^3-6x_1^{13}x_2^4\partial_1^{29}\partial_2^4-5x_1^{16}x_2^2\partial_1^{29}\partial_2^2-6x_1^{14}x_2^4\partial_1^{29}\partial_2^2-4x_1^{15}x_2^2\partial_1^{30}\partial_2^2+$
$6x_1^{14}x_2^3\partial_1^{29}\partial_2^3-x_2^{45}\partial_1^2\partial_2-4x_1^{17}x_2\partial_1^{29}\partial_2+x_1^{15}x_2^3\partial_1^{29}\partial_2+3x_2^{45}\partial_1\partial_2^2+6x_1^{16}x_2^2\partial_1^{28}\partial_2^2+5x_1^{14}x_2^4\partial_1^{28}\partial_2^2+$
$4x_1^{15}x_2^2\partial_1^{29}\partial_2^2+6x_1^{14}x_2^2\partial_1^{30}\partial_2^2+4x_1^{13}x_2^3\partial_1^{29}\partial_2^3+3x_2^{46}\partial_2+6x_1^{16}x_2\partial_1^{29}\partial_2+5x_1^{14}x_2^3\partial_1^{29}\partial_2+3x_2^{45}\partial_2^2+$
$4x_1^{15}x_2^2\partial_1^{28}\partial_2^2-x_1^{13}x_2^4\partial_1^{28}\partial_2^2+3x_1^{14}x_2^2\partial_1^{29}\partial_2^2-2x_1^{16}x_2^2\partial_1^{28}-3x_1^{14}x_2^4\partial_1^{28}-2x_1^{17}\partial_1^{29}-5x_1^{15}x_2^2\partial_1^{29}+$
$5x_2^{45}\partial_2-5x_2^{44}\partial_1\partial_2-2x_1^{16}x_2\partial_1^{28}\partial_2+x_1^{14}x_2^3\partial_1^{28}\partial_2+3x_1^{15}x_2\partial_1^{29}\partial_2+3x_1^{15}x_2^2\partial_1^{27}\partial_2^2+4x_1^{14}x_2^2\partial_1^{28}\partial_2^2+$
$6x_1^{13}x_2^2\partial_1^{29}\partial_2^2+3x_1^{16}\partial_1^{29}+x_1^{14}x_2^2\partial_1^{29}-5x_2^{44}\partial_2+3x_1^{15}x_2\partial_1^{28}\partial_2+5x_1^{13}x_2^3\partial_1^{28}\partial_2+2x_1^{14}x_2\partial_1^{29}\partial_2-$
$x_2^{41}\partial_1^2\partial_2^2+2x_1^{14}x_2^2\partial_1^{27}\partial_2^2-6x_1^{13}x_2^2\partial_1^{28}\partial_2^2+3x_2^{41}\partial_1\partial_2^3+5x_1^{15}x_2^2\partial_1^{27}-x_1^{16}\partial_1^{28}+3x_1^{14}x_2^2\partial_1^{28}-5x_1^{15}\partial_1^{29}-$
$x_1^{15}x_2\partial_1^{27}\partial_2+3x_1^{14}x_2\partial_1^{28}\partial_2+3x_2^{42}\partial_2^2-2x_1^{14}x_2^2\partial_1^{26}\partial_2^2+3x_2^{41}\partial_2^3-5x_1^{15}\partial_1^{28}+x_1^{13}x_2^2\partial_1^{28}+x_1^{14}\partial_1^{29}+$
$6x_2^{40}\partial_1^2\partial_2-5x_1^{14}x_2\partial_1^{27}\partial_2+2x_1^{13}x_2\partial_1^{28}\partial_2+5x_2^{41}\partial_2^2+3x_2^{40}\partial_1\partial_2^2+3x_1^{13}x_2^2\partial_1^{26}\partial_2^2-4x_1^{14}x_2^2\partial_1^{26}+$
$6x_1^{15}\partial_1^{27}-5x_1^{14}\partial_1^{28}-2x_2^{41}\partial_2+5x_1^{14}x_2\partial_1^{26}\partial_2+3x_2^{40}\partial_2^2+4x_1^{14}\partial_1^{27}+x_1^{13}\partial_1^{28}-4x_2^{40}\partial_2-4x_2^{39}\partial_1\partial_2-$
$x_1^{13}x_2\partial_1^{26}\partial_2-4x_1^{14}\partial_1^{26}-4x_2^{39}\partial_2+6x_1^{13}\partial_1^{26}-x_2^{36}\partial_1^2+3x_2^{36}\partial_1\partial_2-2x_1x_2^{32}\partial_1^2\partial_2^3+3x_2^{37}+3x_2^{36}\partial_2+$
$5x_2^{36}-6x_2^{35}\partial_1-4x_2^{35}\partial_2-2x_1x_2^{31}\partial_1^2\partial_2^2-6x_2^{35}-6x_2^{32}\partial_1^2\partial_2+5x_2^{32}\partial_1\partial_2^2-2x_1x_2^{28}\partial_1^2\partial_2^4-2x_1x_2^{15}\partial_1^2\partial_2^{17}+$
$5x_2^{33}\partial_2-6x_1x_2^{30}\partial_1^2\partial_2+5x_2^{32}\partial_2^2+3x_2^{15}\partial_1^2\partial_2^{17}+2x_2^{31}\partial_1^2+4x_2^{32}\partial_2+2x_2^{31}\partial_1\partial_2-4x_2^{31}\partial_2^2-3x_1x_2^{27}\partial_1^2\partial_2^3+$
$5x_1x_2^{14}\partial_1^2\partial_2^{16}-3x_2^{32}+2x_2^{31}\partial_2-5x_2^{28}\partial_1^2\partial_2^2+2x_2^{28}\partial_1\partial_2^3-6x_1^3x_2^{13}\partial_1\partial_2^{15}+4x_1x_2^{15}\partial_1\partial_2^{15}-x_2^{14}\partial_1^2\partial_2^{16}+$

$3x_2^{31}+5x_2^{30}\partial_1-x_2^{30}\partial_2+2x_2^{29}\partial_2^2-5x_1x_2^{26}\partial_1^2\partial_2^2+2x_2^{28}\partial_2^3-4x_1^2x_2^{13}\partial_1\partial_2^{15}+3x_2^{15}\partial_1\partial_2^{15}-4x_1x_2^{13}\partial_1^2\partial_2^{15}+$
$5x_2^{30}+4x_2^{27}\partial_1^2\partial_2-x_2^{28}\partial_2^2+2x_2^{27}\partial_1\partial_2^2+3x_1x_2^{14}\partial_1\partial_2^{14}+x_1^2x_2^{13}\partial_2^{15}-2x_1x_2^{13}\partial_1\partial_2^{15}+6x_2^{13}\partial_1^2\partial_2^{15}+$
$3x_2^{28}\partial_2-4x_1x_2^{25}\partial_1^2\partial_2+2x_2^{27}\partial_2^2-3x_1^4x_2^{17}\partial_1^2\partial_2^6+2x_2^{14}\partial_1\partial_2^{14}+5x_1x_2^{13}\partial_2^{15}+3x_2^{13}\partial_1\partial_2^{15}+6x_2^{27}\partial_2+$
$6x_2^{26}\partial_1\partial_2-2x_1x_2^{23}\partial_1^2\partial_2^2-2x_1^3x_2^{17}\partial_1^2\partial_2^6+x_1^2x_2^{13}\partial_2^{13}-5x_2^{15}\partial_2^{13}-5x_1x_2^{13}\partial_1\partial_2^{13}+6x_2^{26}\partial_2+3x_1^4x_2^{16}\partial_1^2\partial_2^5+$
$4x_1^3x_2^{17}\partial_1\partial_2^6+x_2^{13}\partial_1\partial_2^{13}-4x_2^{26}-5x_1x_2^{22}\partial_1^2\partial_2+4x_1^6x_2^{15}\partial_1\partial_2^4+6x_1^4x_2^{17}\partial_1\partial_2^4+2x_1^3x_2^{16}\partial_1^2\partial_2^5-6x_1^2x_2^{17}\partial_1\partial_2^6-$
$4x_2^{13}\partial_2^{13}-5x_2^{23}\partial_1^2+2x_2^{23}\partial_1\partial_2-2x_1x_2^{19}\partial_1^2\partial_2^3-6x_1^5x_2^{15}\partial_1\partial_2^4-2x_1^3x_2^{17}\partial_1\partial_2^4+3x_1^4x_2^{15}\partial_1^2\partial_2^4-4x_1^3x_2^{16}\partial_1\partial_2^5+$
$x_1x_2^{15}\partial_1^3\partial_2^6+2x_2^{24}+x_1x_2^{21}\partial_1^2+2x_2^{23}\partial_2+3x_1^6x_2^{14}\partial_1\partial_2^3-6x_1^4x_2^{16}\partial_1\partial_2^3-6x_1^5x_2^{15}\partial_2^4+5x_1^3x_2^{17}\partial_2^4-$
$3x_1^4x_2^{15}\partial_1\partial_2^4+2x_1^3x_2^{15}\partial_1^2\partial_2^4-3x_1x_2^{15}\partial_1^4\partial_2^4+6x_1^2x_2^{16}\partial_1\partial_2^5+2x_1x_2^{15}\partial_1^2\partial_2^6+5x_2^{15}\partial_1^3\partial_2^6-x_2^{23}-4x_2^{22}\partial_1-$
$4x_2^{22}\partial_2-x_1x_2^{18}\partial_1^2\partial_2^2+2x_1^5x_2^{14}\partial_1\partial_2^3-2x_1^3x_2^{16}\partial_1\partial_2^3-5x_1^4x_2^{14}\partial_1^2\partial_2^3-4x_1^4x_2^{15}\partial_2^4-x_1^2x_2^{17}\partial_2^4-6x_1^3x_2^{15}\partial_1\partial_2^4-$
$2x_2^{15}\partial_1^4\partial_2^4-5x_1x_2^{14}\partial_1^3\partial_2^5-3x_2^{15}\partial_1^2\partial_2^6-4x_2^{22}-5x_2^{19}\partial_1^2\partial_2-5x_1^5x_2^{15}\partial_2^2-x_1^3x_2^{17}\partial_2^2-5x_1^6x_2^{13}\partial_1\partial_2^2-$
$6x_1^4x_2^{15}\partial_1\partial_2^2+2x_2^{19}\partial_1\partial_2^2+2x_1^5x_2^{14}\partial_2^3-5x_1^3x_2^{16}\partial_2^3+x_1^4x_2^{14}\partial_1\partial_2^3+x_1^3x_2^{14}\partial_1^2\partial_2^3+x_1x_2^{14}\partial_1^4\partial_2^3+4x_1^3x_2^{15}\partial_2^4+$
$6x_1^2x_2^{15}\partial_1\partial_2^4+3x_1^3x_2^{13}\partial_1^2\partial_2^4-2x_1x_2^{15}\partial_1^2\partial_2^4+3x_1x_2^{14}\partial_1^2\partial_2^5+x_2^{14}\partial_1^3\partial_2^5+3x_1x_2^2\partial_1^2\partial_2^{17}-4x_2^{21}+2x_2^{20}\partial_2+$
$3x_1x_2^{17}\partial_1^2\partial_2+2x_2^{19}\partial_2^2+x_1^5x_2^{13}\partial_1\partial_2^2-3x_1^3x_2^{15}\partial_1\partial_2^2+x_1^4x_2^{13}\partial_1^2\partial_2^2+4x_1^3x_2^{13}\partial_1^3\partial_2^2+6x_1x_2^{15}\partial_1^3\partial_2^2-$
$3x_1^4x_2^{14}\partial_2^3+x_1^2x_2^{16}\partial_2^3+3x_1^3x_2^{14}\partial_1\partial_2^3+5x_2^{14}\partial_1^4\partial_2^3-6x_1^2x_2^{15}\partial_2^4+6x_1^3x_2^{13}\partial_1\partial_2^4-4x_1x_2^{15}\partial_1\partial_2^4+2x_1^2x_2^{13}\partial_1^2\partial_2^4+$
$5x_2^{15}\partial_1^2\partial_2^4-x_1x_2^{13}\partial_1^3\partial_2^4+2x_2^{14}\partial_1^2\partial_2^5+2x_2^2\partial_1^2\partial_2^{17}-3x_2^{18}\partial_1^2-4x_1^3x_2^{16}\partial_2-x_2^{19}\partial_2-3x_1^4x_2^{14}\partial_1\partial_2+$
$5x_2^{18}\partial_1\partial_2+x_1^5x_2^{13}\partial_2^2+2x_1^3x_2^{15}\partial_2^2-6x_1^4x_2^{13}\partial_1\partial_2^2+5x_1^3x_2^{13}\partial_1^2\partial_2^2-6x_1^2x_2^{13}\partial_1^3\partial_2^2-2x_2^{15}\partial_1^3\partial_2^2-6x_1x_2^{13}\partial_1^4\partial_2^2+$
$3x_1^3x_2^{14}\partial_2^3+3x_1^2x_2^{14}\partial_1\partial_2^3-3x_1x_2^{14}\partial_1^2\partial_2^3+3x_1^2x_2^{13}\partial_1\partial_2^4-3x_2^{15}\partial_1\partial_2^4-x_1x_2^{13}\partial_1^2\partial_2^4-5x_2^{13}\partial_1^3\partial_2^4-$
$x_1x_2\partial_1^2\partial_2^{16}-2x_2^{19}-4x_1x_2^{16}\partial_1^2+5x_2^{18}\partial_2-2x_1^3x_2^{14}\partial_1\partial_2-2x_1x_2^{14}\partial_1^3\partial_2+5x_1^4x_2^{13}\partial_2^2+x_1^2x_2^{15}\partial_2^2-$
$x_1^3x_2^{13}\partial_1\partial_2^2-2x_1^2x_2^{13}\partial_1^2\partial_2^2-3x_1x_2^{13}\partial_1^3\partial_2^2-4x_2^{13}\partial_1^4\partial_2^2+2x_1^2x_2^{14}\partial_2^3-6x_1x_2^{14}\partial_1\partial_2^3+6x_2^{14}\partial_1^2\partial_2^3-$
$x_1^2x_2^{13}\partial_2^4-3x_1x_2^{13}\partial_1\partial_2^4-5x_2^{13}\partial_1^2\partial_2^4-4x_1^3\partial_1\partial_2^{15}-6x_1x_2^2\partial_1\partial_2^{15}-5x_2\partial_1^2\partial_2^{16}-2x_1^3x_2^{15}+2x_2^{18}-$
$2x_1^4x_2^{13}\partial_1-x_2^{17}\partial_1+4x_1^3x_2^{14}\partial_2+3x_2^{14}\partial_1^3\partial_2-5x_1^3x_2^{13}\partial_2^2-5x_1^2x_2^{13}\partial_1\partial_2^2-4x_2^{15}\partial_1\partial_2^2+5x_1x_2^{13}\partial_1^2\partial_2^2-$
$2x_2^{13}\partial_1^3\partial_2^2-x_2^{14}\partial_1\partial_2^3+x_1x_2^{13}\partial_2^4-3x_2^{13}\partial_1\partial_2^4+6x_1^2\partial_1\partial_2^{15}+2x_2^2\partial_1\partial_2^{15}+6x_1\partial_1^2\partial_2^{15}-x_2^{17}+3x_1^3x_2^{13}\partial_1-$
$5x_1^2x_2^{13}\partial_1^2-x_2^{15}\partial_1^2-x_1x_2^{13}\partial_1^3-6x_1^2x_2^{14}\partial_2+5x_2^{15}\partial_2^2+2x_1x_2^{13}\partial_1\partial_2^2+x_2^{13}\partial_1^2\partial_2^2+4x_2^{13}\partial_2^4+2x_1x_2\partial_1\partial_2^{14}+$
$5x_1^2\partial_2^{15}+3x_1\partial_1\partial_2^{15}+4\partial_1^2\partial_2^{15}-6x_1^3x_2^{13}-5x_2^{13}\partial_1^3-3x_2^{14}\partial_1\partial_2-x_1x_2^{13}\partial_2^2-6x_2^{13}\partial_1\partial_2^2-2x_1^4x_2^4\partial_1^2\partial_2^6-$
$3x_2\partial_1\partial_2^{14}-x_1\partial_2^{15}+2\partial_1\partial_2^{15}-4x_1^2x_2^{13}+6x_1x_2^{13}\partial_1-6x_2^{13}\partial_1^2-6x_2^{14}\partial_2+4x_2^{13}\partial_2^2+3x_1^3x_2^4\partial_1^2\partial_2^6+$
$5x_1^2\partial_2^{13}+x_2^2\partial_2^{13}+x_1\partial_1\partial_2^{13}+5x_2^{13}\partial_1-5x_1^2x_2^4\partial_1^4\partial_2^4+2x_1^4x_2^3\partial_1^2\partial_2^5-6x_1^3x_2^4\partial_1\partial_2^6+5\partial_1\partial_2^{13}-6x_1^6x_2^2\partial_1\partial_2^4+$
$4x_1^4x_2^4\partial_1\partial_2^4+x_1x_2^4\partial_1^4\partial_2^4-3x_1^3x_2^3\partial_1^2\partial_2^5-4x_1^2x_2^4\partial_1\partial_2^6+6\partial_2^{13}-x_1^2x_2^3\partial_1^4\partial_2^3-4x_1^5x_2^2\partial_1\partial_2^4+3x_1^3x_2^4\partial_1\partial_2^4+$
$2x_1^4x_2^2\partial_1^2\partial_2^4-5x_1x_2^4\partial_1^3\partial_2^4+6x_1^3x_2^3\partial_1\partial_2^5+5x_1x_2^2\partial_1^3\partial_2^6-2x_1^4x_2^2\partial_1^3\partial_2^2-3x_1^2x_2^4\partial_1^3\partial_2^2+2x_1^6x_2\partial_1\partial_2^3-$
$4x_1^4x_2^3\partial_1\partial_2^3-5x_1x_2^3\partial_1^4\partial_2^3-4x_1^5x_2^2\partial_2^4-x_1^3x_2^4\partial_2^4-2x_1^4x_2^2\partial_1\partial_2^4-3x_1^3x_2^2\partial_1^2\partial_2^4+x_2^4\partial_1^3\partial_2^4-2x_1x_2^2\partial_1^4\partial_2^4+$
$4x_1^2x_2^3\partial_1\partial_2^5-3x_1x_2^2\partial_1^2\partial_2^6-x_2^2\partial_1^3\partial_2^6+3x_1^3x_2^2\partial_1^3\partial_2^2+x_1x_2^4\partial_1^3\partial_2^2+5x_1^2x_2^2\partial_1^4\partial_2^2-3x_1^5x_2\partial_1\partial_2^3+3x_1^3x_2^3\partial_1\partial_2^3+$
$x_1^4x_2\partial_1^2\partial_2^3-x_1x_2^3\partial_1^3\partial_2^3+6x_1^4x_2^2\partial_2^4-5x_1^2x_2^4\partial_2^4-4x_1^3x_2^2\partial_1\partial_2^4+3x_2^2\partial_1^4\partial_2^4+x_1x_2\partial_1^3\partial_2^5-2x_2^2\partial_1^2\partial_2^6+$
$5x_1^4x_2\partial_1^3\partial_2+2x_1^2x_2^3\partial_1^3\partial_2+x_1^5x_2^2\partial_2^2-5x_1^3x_2^4\partial_2^2+x_1^6\partial_1\partial_2^2-4x_1^4x_2^2\partial_1\partial_2^2-x_1^3x_2^2\partial_1^2\partial_2^2-3x_1x_2^4\partial_1^2\partial_2^2-$

$5x_1^2x_2^2\partial_1^3\partial_2^2 - x_1x_2^2\partial_1^4\partial_2^2 - 3x_1^5x_2\partial_2^3 + x_1^3x_2^3\partial_2^3 + 5x_1^4x_2\partial_1\partial_2^3 + 5x_1^3x_2\partial_1^2\partial_2^3 - 5x_2^3\partial_1^3\partial_2^3 + 5x_1x_2\partial_1^4\partial_2^3 -$
$6x_1^3x_2^2\partial_2^4 + 4x_1^2x_2^2\partial_1\partial_2^4 + 2x_1^3\partial_1^2\partial_2^4 + x_1x_2^2\partial_1^2\partial_2^4 + 2x_1x_2\partial_1^2\partial_2^5 + 5x_2\partial_1^3\partial_2^5 - x_1^3x_2\partial_1^3\partial_2 - 3x_1x_2^3\partial_1^3\partial_2 +$
$5x_1^5\partial_1\partial_2^2 - 2x_1^3x_2^2\partial_1\partial_2^2 + 5x_1^4\partial_1^2\partial_2^2 - 5x_1^2x_2^2\partial_1^2\partial_2^2 - 2x_2^4\partial_1^2\partial_2^2 - 6x_1^3\partial_1^3\partial_2^2 - 3x_1x_2^2\partial_1^3\partial_2^2 - 2x_1^4x_2\partial_2^3 +$
$5x_1^2x_2^3\partial_2^3 + 2x_1^3x_2\partial_1\partial_2^3 - x_2\partial_1^4\partial_2^3 - 4x_1^2x_2^2\partial_2^4 + 4x_1^3\partial_1\partial_2^4 + 6x_1x_2^2\partial_1\partial_2^4 - 3x_1^2\partial_1^2\partial_2^4 + 2x_2^2\partial_1^2\partial_2^4 -$
$5x_1\partial_1^3\partial_2^4 - 3x_2\partial_1^2\partial_2^5 - 4x_1^3x_2^2\partial_1^2 - 6x_1x_2^4\partial_1^2 - 4x_1^4\partial_1^3 + 3x_1^2x_2^2\partial_1^3 + 6x_1^3x_2^3\partial_2 - 2x_1^4x_2\partial_1\partial_2 - 4x_1^3x_2\partial_1^2\partial_2 +$
$2x_1x_2^3\partial_1^2\partial_2 + 6x_1^2x_2\partial_1^3\partial_2 + 5x_1^5\partial_2^2 - 3x_1^3x_2^2\partial_2^2 - 4x_1^4\partial_1\partial_2^2 + 6x_1^2x_2^2\partial_1\partial_2^2 - x_1^3\partial_1^2\partial_2^2 - 5x_1x_2^2\partial_1^2\partial_2^2 -$
$4x_1^2\partial_1^3\partial_2^2 + 2x_2^2\partial_1^3\partial_2^2 - 4x_1\partial_1^4\partial_2^2 + 2x_1^3x_2\partial_2^3 + 2x_1^2x_2\partial_1\partial_2^3 + 3x_1x_2\partial_1^2\partial_2^3 + 2x_1^2\partial_1\partial_2^4 - 2x_2^2\partial_1\partial_2^4 -$
$5x_1\partial_1^2\partial_2^4 + \partial_1^3\partial_2^4 + 6x_1^3\partial_1^3 + 2x_1x_2^2\partial_1^3 + 3x_1^3x_2\partial_1\partial_2 + 6x_1^2x_2\partial_1^2\partial_2 - 3x_2^3\partial_1^2\partial_2 - 6x_1x_2\partial_1^3\partial_2 - x_1^4\partial_2^2 +$
$5x_1^2x_2^2\partial_2^2 + 2x_1^3\partial_1\partial_2^2 - 5x_1x_2^2\partial_1\partial_2^2 + 3x_1^2\partial_1^2\partial_2^2 + x_2^2\partial_1^2\partial_2^2 - 2x_1\partial_1^3\partial_2^2 + 6\partial_1^4\partial_2^2 - 3x_1^2x_2\partial_2^3 - 4x_1x_2\partial_1\partial_2^3 +$
$3x_2\partial_1^2\partial_2^3 - 5x_1^2\partial_2^4 - 2x_1\partial_1\partial_2^4 + \partial_1^2\partial_2^4 + 3x_1^3x_2^2 + 3x_1^4\partial_1 - 3x_1^2x_2^2\partial_1 - 2x_1^3\partial_1^2 + 6x_1x_2^2\partial_1^2 + 3x_1^2\partial_1^3 -$
$6x_1^3x_2\partial_2 - 2x_1^2x_2\partial_1\partial_2 + 6x_1x_2\partial_1^2\partial_2 + 2x_2\partial_1^3\partial_2 + x_1^3\partial_2^2 - 4x_1x_2^2\partial_2^2 - 3x_1^2\partial_1\partial_2^2 - 4x_2^2\partial_1\partial_2^2 - 5x_1\partial_1^2\partial_2^2 +$
$3\partial_1^3\partial_2^2 - 5x_2\partial_1\partial_2^3 + 5x_1\partial_2^4 - 2\partial_1\partial_2^4 + 2x_1^3\partial_1 + 4x_1^2\partial_1^2 - 3x_2^2\partial_1^2 - 3x_1\partial_1^3 - 4x_1^2x_2\partial_2 + 6x_1x_2\partial_1\partial_2 +$
$4x_2\partial_1^2\partial_2 + x_1^2\partial_2^2 + 5x_2^2\partial_2^2 - 5x_1\partial_1\partial_2^2 - 2\partial_1^2\partial_2^2 - 6\partial_2^4 - 4x_1^3 + 5x_1x_2^2 - x_1^2\partial_1 + 3x_1\partial_1^2 + \partial_1^3 -$
$3x_1x_2\partial_2 - \partial_1\partial_2^2 - 6x_1^2 - 5x_2^2 - 6x_1\partial_1 - 2\partial_1^2 - 6x_2\partial_2 - 6\partial_2^2 + 5x_1 - 5$

(3) **Example 6.4.3**

The polynomials $p_1, p_2, p_3 \in A_3 = \mathbb{F}_2[x_1, x_2, x_3, \partial_1, \partial_2, \partial_3]$ of the public key Q are:

$p_1 = x_1^{11}x_3^{15}\partial_1^5\partial_3^3 + x_1^{12}x_3^{14}\partial_1^4\partial_3^2 + x_1^{11}x_3^{14}\partial_1^5\partial_3^2 + x_1^{10}x_3^{15}\partial_1^4\partial_3^3 + x_1^{11}x_2^4x_3^9\partial_1^5\partial_3^3 + x_1^{11}x_3^{13}\partial_1^5\partial_3^3 +$
$x_1^9x_3^{15}\partial_1^5\partial_3^3 + x_1^{14}x_3^{10}\partial_1^4\partial_3^2 + x_1^{10}x_3^{14}\partial_1^4\partial_3^2 + x_1^{11}x_2^4x_3^8\partial_1^5\partial_3^2 + x_1^{11}x_3^{12}\partial_1^5\partial_3^2 + x_1^9x_3^{14}\partial_1^5\partial_3^2 + x_1^{10}x_2^4x_3^9\partial_1^4\partial_3^3 +$
$x_1^{10}x_3^{13}\partial_1^4\partial_3^3 + x_1^8x_3^{15}\partial_1^4\partial_3^3 + x_1^{19}x_3^3\partial_1^5\partial_3^3 + x_1^{15}x_2^4x_3^3\partial_1^5\partial_3^3 + x_1^{15}x_2^2x_3^5\partial_1^5\partial_3^3 + x_1^{13}x_2^4x_3^5\partial_1^5\partial_3^3 + x_1^{15}x_3^7\partial_1^5\partial_3^3 +$
$x_1^{13}x_2^2x_3^7\partial_1^5\partial_3^3 + x_1^9x_2^4x_3^9\partial_1^5\partial_3^3 + x_1^9x_3^{13}\partial_1^5\partial_3^3 + x_1^{12}x_3^{13}\partial_1^2\partial_3^2 + x_1^{11}x_3^{13}\partial_1^2\partial_3^2 + x_1^5x_2^4x_3^{15}\partial_1^2\partial_3^2 + x_1^5x_2^2x_3^{17}\partial_1^2\partial_3^2 +$
$x_1^{10}x_2^4x_3^8\partial_1^4\partial_3^2 + x_1^{10}x_3^{12}\partial_1^4\partial_3^2 + x_1^8x_3^{14}\partial_1^4\partial_3^2 + x_1^{19}x_3^2\partial_1^5\partial_3^2 + x_1^{15}x_2^4x_3^2\partial_1^5\partial_3^2 + x_1^{15}x_2^2x_3^4\partial_1^5\partial_3^2 + x_1^{13}x_2^4x_3^4\partial_1^5\partial_3^2 +$
$x_1^{15}x_3^6\partial_1^5\partial_3^2 + x_1^{13}x_2^2x_3^6\partial_1^5\partial_3^2 + x_1^9x_2^4x_3^8\partial_1^5\partial_3^2 + x_1^9x_3^{12}\partial_1^5\partial_3^2 + x_1^{18}x_3^3\partial_1^4\partial_3^3 + x_1^{14}x_2^4x_3^3\partial_1^4\partial_3^3 + x_1^{14}x_2^2x_3^5\partial_1^4\partial_3^3 +$
$x_1^{12}x_2^4x_3^5\partial_1^4\partial_3^3 + x_1^{14}x_3^7\partial_1^4\partial_3^3 + x_1^{12}x_2^2x_3^7\partial_1^4\partial_3^3 + x_1^8x_2^4x_3^9\partial_1^4\partial_3^3 + x_1^8x_3^{13}\partial_1^4\partial_3^3 + x_1^{15}x_2^4x_3\partial_1^5\partial_3^3 + x_1^{15}x_2^2x_3^3\partial_1^5\partial_3^3 +$
$x_1^{15}x_3^5\partial_1^5\partial_3^3 + x_1^9x_2^4x_3^7\partial_1^5\partial_3^3 + x_1^{11}x_3^9\partial_1^5\partial_3^3 + x_1^9x_2^2x_3^9\partial_1^5\partial_3^3 + x_1^{14}x_3^9\partial_1^2\partial_3^2 + x_1^{10}x_3^{13}\partial_1^2\partial_3^2 + x_1^{11}x_3^{12}\partial_1^2\partial_3 +$
$x_1^5x_2^4x_3^{14}\partial_1^2\partial_3 + x_1^5x_2^2x_3^{16}\partial_1^2\partial_3 + x_1^4x_2^4x_3^{14}\partial_1^3\partial_3 + x_1^4x_2^2x_3^{16}\partial_1^3\partial_3 + x_1^{10}x_3^{13}\partial_1\partial_3^2 + x_1^4x_2^4x_3^{15}\partial_1\partial_3^2 + x_1^4x_2^2x_3^{17}\partial_1\partial_3^2 +$
$x_1^3x_2^4x_3^{15}\partial_1^2\partial_3^2 + x_1^3x_2^2x_3^{17}\partial_1^2\partial_3^2 + x_1^{14}x_3^6\partial_1^4\partial_3^2 + x_1^{12}x_2^2x_3^6\partial_1^4\partial_3^2 + x_1^{12}x_3^8\partial_1^4\partial_3^2 + x_1^8x_2^4x_3^8\partial_1^4\partial_3^2 + x_1^8x_3^{12}\partial_1^4\partial_3^2 +$
$x_1^{15}x_2^4\partial_1^5\partial_3^2 + x_1^{15}x_2^2x_3^2\partial_1^5\partial_3^2 + x_1^{15}x_3^4\partial_1^5\partial_3^2 + x_1^9x_2^4x_3^6\partial_1^5\partial_3^2 + x_1^{11}x_3^8\partial_1^5\partial_3^2 + x_1^9x_2^2x_3^8\partial_1^5\partial_3^2 + x_1^{14}x_2^4x_3\partial_1^4\partial_3^3 +$
$x_1^{14}x_2^2x_3^3\partial_1^4\partial_3^3 + x_1^{14}x_3^5\partial_1^4\partial_3^3 + x_1^8x_2^4x_3^7\partial_1^4\partial_3^3 + x_1^{10}x_3^9\partial_1^4\partial_3^3 + x_1^8x_2^2x_3^9\partial_1^4\partial_3^3 + x_1^{13}x_2^4x_3\partial_1^5\partial_3^3 + x_1^{15}x_3^3\partial_1^5\partial_3^3 +$
$x_1^{13}x_2^2x_3^3\partial_1^5\partial_3^3 + x_1^{13}x_3^5\partial_1^5\partial_3^3 + x_1^{11}x_2^2x_3^5\partial_1^5\partial_3^3 + x_1^9x_3^9\partial_1^5\partial_3^3 + x_1^{10}x_2^4x_3^7\partial_1^2\partial_3^2 + x_1^{10}x_3^{11}\partial_1^2\partial_3^2 + x_1^8x_3^{13}\partial_1^2\partial_3^2 +$
$x_1^{10}x_3^{12}\partial_1\partial_3 + x_1^4x_2^4x_3^{14}\partial_1\partial_3 + x_1^4x_2^2x_3^{16}\partial_1\partial_3 + x_1^3x_2^4x_3^{14}\partial_1^2\partial_3 + x_1^3x_2^2x_3^{16}\partial_1^2\partial_3 + x_1^{12}x_3^8\partial_1^3\partial_3 + x_1^8x_3^{12}\partial_1^3\partial_3 +$
$x_1^2x_2^4x_3^{14}\partial_1^3\partial_3 + x_1^2x_2^2x_3^{16}\partial_1^3\partial_3 + x_1^2x_2^4x_3^{15}\partial_1\partial_3^2 + x_1^2x_2^2x_3^{17}\partial_1\partial_3^2 + x_1^3x_2^4x_3^{13}\partial_1^2\partial_3^2 + x_1^5x_3^{15}\partial_1^2\partial_3^2 + x_1^3x_2^2x_3^{15}\partial_1^2\partial_3^2 +$

$$x_1^{14}x_3^4\partial_1^4\partial_3^2+x_1^8x_2^4x_3^6\partial_1^4\partial_3^2+x_1^{10}x_3^8\partial_1^4\partial_3^2+x_1^8x_2^2x_3^8\partial_1^4\partial_3^2+x_1^{13}x_2^4\partial_1^5\partial_3^2+x_1^{15}x_3^2\partial_1^5\partial_3^2+x_1^{13}x_2^2x_3^2\partial_1^5\partial_3^2+$$
$$x_1^{13}x_3^4\partial_1^5\partial_3^2+x_1^{11}x_2^2x_3^4\partial_1^5\partial_3^2+x_1^9x_3^8\partial_1^5\partial_3^2+x_1^{12}x_2^4x_3\partial_1^4\partial_3^3+x_1^{14}x_3^3\partial_1^4\partial_3^3+x_1^{12}x_2^2x_3^3\partial_1^4\partial_3^3+x_1^{12}x_3^5\partial_1^4\partial_3^3+$$
$$x_1^{10}x_2^2x_3^5\partial_1^4\partial_3^3+x_1^8x_3^9\partial_1^4\partial_3^3+x_1^{15}x_3\partial_1^5\partial_3^3+x_1^{13}x_2^2x_3\partial_1^5\partial_3^3+x_1^{13}x_3^3\partial_1^5\partial_3^3+x_1^9x_2^2x_3^5\partial_1^5\partial_3^3+x_1^9x_3^7\partial_1^5\partial_3^3+$$
$$x_1^{14}x_3^5\partial_1^2\partial_3^2+x_1^{12}x_2^2x_3^5\partial_1^2\partial_3^2+x_1^{12}x_3^7\partial_1^2\partial_3^2+x_1^8x_2^4x_3^7\partial_1^2\partial_3^2+x_1^8x_3^{11}\partial_1^2\partial_3^2+x_1^2x_2^4x_3^{14}\partial_1\partial_3+x_1^2x_2^2x_3^{16}\partial_1\partial_3+$$
$$x_1^3x_2^4x_3^{12}\partial_1^2\partial_3+x_1^5x_3^{14}\partial_1^2\partial_3+x_1^3x_2^2x_3^{14}\partial_1^2\partial_3+x_1^8x_2^4x_3^6\partial_1^3\partial_3+x_1^8x_3^{10}\partial_1^3\partial_3+x_1^6x_3^{12}\partial_1^3\partial_3+x_1^2x_2^4x_3^{12}\partial_1^3\partial_3+$$
$$x_1^4x_3^{14}\partial_1^3\partial_3+x_1^2x_2^2x_3^{14}\partial_1^3\partial_3+x_1^2x_2^4x_3^{13}\partial_1\partial_3^2+x_1^4x_3^{15}\partial_1\partial_3^2+x_1^2x_2^2x_3^{15}\partial_1\partial_3^2+x_1^5x_3^{13}\partial_1^2\partial_3^2+x_1^3x_2^2x_3^{13}\partial_1^2\partial_3^2+$$
$$x_1^{10}x_2^2x_3^4\partial_1^4\partial_3^2+x_1^8x_3^8\partial_1^4\partial_3^2+x_1^{15}\partial_1^5\partial_3^2+x_1^{13}x_2^2\partial_1^5\partial_3^2+x_1^{13}x_3^2\partial_1^5\partial_3^2+x_1^9x_2^2x_3^4\partial_1^5\partial_3^2+x_1^9x_3^6\partial_1^5\partial_3^2+$$
$$x_1^{14}x_3\partial_1^4\partial_3^3+x_1^{12}x_2^2x_3\partial_1^4\partial_3^3+x_1^{12}x_3^3\partial_1^4\partial_3^3+x_1^8x_2^2x_3^5\partial_1^4\partial_3^3+x_1^8x_3^7\partial_1^4\partial_3^3+x_1^{13}x_3\partial_1^5\partial_3^3+x_1^{14}x_3^3\partial_1^2\partial_3^2+$$
$$x_1^8x_2^4x_3^5\partial_1^2\partial_3^2+x_1^{10}x_3^7\partial_1^2\partial_3^2+x_1^8x_2^2x_3^7\partial_1^2\partial_3^2+x_1^2x_2^4x_3^{12}\partial_1\partial_3+x_1^4x_3^{14}\partial_1\partial_3+x_1^2x_2^2x_3^{14}\partial_1\partial_3+x_1^5x_3^{12}\partial_1^2\partial_3+$$
$$x_1^3x_2^2x_3^{12}\partial_1^2\partial_3+x_1^{12}x_3^4\partial_1^3\partial_3+x_1^{10}x_2^2x_3^4\partial_1^3\partial_3+x_1^{10}x_3^6\partial_1^3\partial_3+x_1^6x_2^4x_3^6\partial_1^3\partial_3+x_1^6x_3^{10}\partial_1^3\partial_3+x_1^4x_3^{12}\partial_1^3\partial_3+$$
$$x_1^2x_2^2x_3^{12}\partial_1^3\partial_3+x_1^4x_3^{13}\partial_1\partial_3^2+x_1^2x_2^2x_3^{13}\partial_1\partial_3^2+x_1^8x_2^2x_3^4\partial_1^4\partial_3^2+x_1^8x_3^6\partial_1^4\partial_3^2+x_1^{13}\partial_1^5\partial_3^2+x_1^{12}x_3\partial_1^4\partial_3^3+$$
$$x_1^3x_2^4x_3^{12}+x_1^3x_2^2x_3^{14}+x_1^{10}x_2^2x_3^3\partial_1^2\partial_3^2+x_1^8x_3^7\partial_1^2\partial_3^2+x_1^4x_3^{12}\partial_1\partial_3+x_1^2x_2^2x_3^{12}\partial_1\partial_3+x_1^{12}x_3^2\partial_1^3\partial_3+x_1^6x_2^4x_3^4\partial_1^3\partial_3+$$
$$x_1^8x_3^6\partial_1^3\partial_3+x_1^6x_2^2x_3^6\partial_1^3\partial_3+x_1^{11}x_3^6+x_1^7x_3^{10}+x_1x_2^4x_3^{12}+x_1x_2^2x_3^{14}+x_1^8x_2^2x_3^3\partial_1^2\partial_3^2+x_1^8x_3^5\partial_1^2\partial_3^2+x_1^8x_2^2x_3^2\partial_1^3\partial_3+$$
$$x_1^6x_3^6\partial_1^3\partial_3+x_1^7x_2^4x_3^4+x_1^7x_3^8+x_1^5x_3^{10}+x_1x_2^4x_3^{10}+x_1^3x_3^{12}+x_1x_2^2x_3^{12}+x_1^6x_2^2x_3^2\partial_1^3\partial_3+x_1^6x_3^4\partial_1^3\partial_3+$$
$$x_1^{11}x_3^2+x_1^9x_2^2x_3^2+x_1^9x_3^4+x_1^5x_2^4x_3^4+x_1^5x_3^8+x_1^3x_3^{10}+x_1x_2^2x_3^{10}+x_1^{11}+x_1^5x_2^4x_3^2+x_1^7x_3^4+x_1^5x_2^2x_3^4+$$
$$x_1^7x_2^2+x_1^5x_3^4+x_1^5x_2^2+x_1^5x_3^2,$$
$$p_2=x_1^3x_2^4x_3^{14}\partial_1^3\partial_2^3\partial_3+x_1^3x_2^3x_3^{14}\partial_1^3\partial_2^3\partial_3+x_1^3x_2^7x_3^{10}\partial_1^3\partial_2^2\partial_3+x_1^3x_2^5x_3^{12}\partial_1^3\partial_2^2\partial_3+x_1^2x_2^8x_3^{10}\partial_1^2\partial_2^3\partial_3+$$
$$x_1^2x_2^6x_3^{12}\partial_1^2\partial_2^3\partial_3+x_1^2x_2^4x_3^{14}\partial_1^2\partial_2^3\partial_3+x_1^5x_2^8x_3^6\partial_1^3\partial_2^3\partial_3+x_1^5x_2^6x_3^8\partial_1^3\partial_2^3\partial_3+x_1^3x_2^6x_3^{10}\partial_1^3\partial_2^2\partial_3+x_1^3x_2^4x_3^{12}\partial_1^3\partial_2^2\partial_3+$$
$$x_1^3x_2^2x_3^{14}\partial_1^3\partial_2^2\partial_3+x_1^2x_2^7x_3^{10}\partial_1^2\partial_2^3\partial_3+x_1^2x_2^5x_3^{12}\partial_1^2\partial_2^3\partial_3+x_1^2x_2^3x_3^{14}\partial_1^2\partial_2^3\partial_3+x_1^5x_2^7x_3^6\partial_1^3\partial_2^3\partial_3+x_1^5x_2^5x_3^8\partial_1^3\partial_2^3\partial_3+$$
$$x_1^4x_2^7x_3^{12}\partial_3+x_1^4x_2^5x_3^{14}\partial_3+x_1^4x_2^8x_3^6\partial_1^2\partial_2^3\partial_3+x_1^4x_2^6x_3^8\partial_1^2\partial_2^3\partial_3+x_1^3x_2^8x_3^6\partial_1^3\partial_2^3\partial_3+x_1^3x_2^6x_3^8\partial_1^3\partial_2^3\partial_3+$$
$$x_1^3x_2^4x_3^{10}\partial_1^3\partial_2^3\partial_3+x_1^6x_2^3x_3^{12}\partial_1\partial_2+x_1^6x_2x_3^{14}\partial_1\partial_2+x_1^4x_2^6x_3^{12}\partial_3+x_1^4x_2^4x_3^{14}\partial_3+x_1^2x_2^6x_3^{10}\partial_1^2\partial_2^2\partial_3+x_1^2x_2^4x_3^{12}\partial_1^2\partial_2^2\partial_3+$$
$$x_1^2x_2^2x_3^{14}\partial_1^2\partial_2^2\partial_3+x_1^5x_2^6x_3^6\partial_1^3\partial_2^2\partial_3+x_1^5x_2^4x_3^8\partial_1^3\partial_2^2\partial_3+x_1^4x_2^7x_3^6\partial_1^2\partial_2^3\partial_3+x_1^4x_2^5x_3^8\partial_1^2\partial_2^3\partial_3+x_1^3x_2^7x_3^6\partial_1^3\partial_2^3\partial_3+$$
$$x_1^3x_2^5x_3^8\partial_1^3\partial_2^3\partial_3+x_1^3x_2^3x_3^{10}\partial_1^3\partial_2^3\partial_3+x_1^6x_2^2x_3^{12}\partial_1\partial_2+x_1^6x_3^{14}\partial_1\partial_2+x_1^6x_2^9x_3^6\partial_3+x_1^6x_2^7x_3^8\partial_3+x_1^2x_2^8x_3^6\partial_1^2\partial_2^3\partial_3+$$
$$x_1^2x_2^6x_3^8\partial_1^2\partial_2^3\partial_3+x_1^2x_2^4x_3^{10}\partial_1^2\partial_2^3\partial_3+x_1^3x_2^8x_3^4\partial_1^3\partial_2^3\partial_3+x_1^5x_2^4x_3^6\partial_1^3\partial_2^3\partial_3+x_1^3x_2^4x_3^8\partial_1^3\partial_2^3\partial_3+x_1^6x_2^3x_3^{12}+$$
$$x_1^6x_2x_3^{14}+x_1^6x_2^2x_3^{12}\partial_1+x_1^6x_3^{14}\partial_1+x_1^8x_2^5x_3^6\partial_1\partial_2+x_1^8x_2^3x_3^8\partial_1\partial_2+x_1^6x_2^8x_3^6\partial_3+x_1^6x_2^4x_3^{10}\partial_3+x_1^4x_2^6x_3^6\partial_1^2\partial_2^2\partial_3+$$
$$x_1^4x_2^4x_3^8\partial_1^2\partial_2^2\partial_3+x_1^3x_2^6x_3^6\partial_1^3\partial_2^2\partial_3+x_1^3x_2^4x_3^8\partial_1^3\partial_2^2\partial_3+x_1^3x_2^2x_3^{10}\partial_1^3\partial_2^2\partial_3+x_1^2x_2^7x_3^6\partial_1^2\partial_2^3\partial_3+x_1^2x_2^5x_3^8\partial_1^2\partial_2^3\partial_3+$$
$$x_1^2x_2^3x_3^{10}\partial_1^2\partial_2^3\partial_3+x_1^3x_2^7x_3^4\partial_1^3\partial_2^3\partial_3+x_1^5x_2^3x_3^6\partial_1^3\partial_2^3\partial_3+x_1^3x_2^3x_3^8\partial_1^3\partial_2^3\partial_3+x_1^6x_2^2x_3^{12}+x_1^6x_3^{14}+x_1^8x_2^4x_3^6\partial_1\partial_2+$$
$$x_1^8x_2^2x_3^8\partial_1\partial_2+x_1^4x_2^9x_3^6\partial_3+x_1^6x_2^5x_3^8\partial_3+x_1^4x_2^7x_3^8\partial_3+x_1^6x_2^3x_3^{10}\partial_3+x_1^4x_2^5x_3^{10}\partial_3+x_1^3x_2^5x_3^6\partial_1^3\partial_2^2\partial_3+x_1^3x_2^3x_3^8\partial_1^3\partial_2^2\partial_3+$$
$$x_1^2x_2^8x_3^4\partial_1^2\partial_2^3\partial_3+x_1^4x_2^4x_3^6\partial_1^2\partial_2^3\partial_3+x_1^2x_2^6x_3^6\partial_1^2\partial_2^3\partial_3+x_1^5x_2^6x_3^2\partial_1^3\partial_2^3\partial_3+x_1^5x_2^4x_3^4\partial_1^3\partial_2^3\partial_3+x_1^3x_2^4x_3^6\partial_1^3\partial_2^3\partial_3+$$
$$x_1^8x_2^5x_3^6+x_1^8x_2^3x_3^8+x_1^8x_2^4x_3^6\partial_1+x_1^8x_2^2x_3^8\partial_1+x_1^6x_2^5x_3^6\partial_1\partial_2+x_1^6x_2^3x_3^8\partial_1\partial_2+x_1^6x_2x_3^{10}\partial_1\partial_2+x_1^8x_2^8x_3^2\partial_3+$$
$$x_1^8x_2^6x_3^4\partial_3+x_1^4x_2^8x_3^6\partial_3+x_1^2x_2^{10}x_3^6\partial_3+x_1^4x_2^6x_3^8\partial_3+x_1^2x_2^8x_3^8\partial_3+x_1^4x_2^4x_3^{10}\partial_3+x_1^2x_2^6x_3^6\partial_1^2\partial_2^2\partial_3+x_1^2x_2^4x_3^8\partial_1^2\partial_2^2\partial_3+$$
$$x_1^2x_2^2x_3^{10}\partial_1^2\partial_2^2\partial_3+x_1^3x_2^6x_3^4\partial_1^3\partial_2^2\partial_3+x_1^5x_2^2x_3^6\partial_1^3\partial_2^2\partial_3+x_1^3x_2^4x_3^6\partial_1^3\partial_2^2\partial_3+x_1^2x_2^7x_3^4\partial_1^2\partial_2^3\partial_3+x_1^4x_2^3x_3^6\partial_1^2\partial_2^3\partial_3+$$

$x_1^2x_2^5x_3^6\partial_1^2\partial_2^3\partial_3+x_1^5x_2^5x_3^2\partial_1^3\partial_2^3\partial_3+x_1^5x_2^3x_3^4\partial_1^3\partial_2^3\partial_3+x_1^3x_2^3x_3^6\partial_1^3\partial_2^3\partial_3+x_1^8x_2^4x_3^6+x_1^8x_2^2x_3^8+x_1^6x_2^4x_3^6\partial_1\partial_2+$
$x_1^6x_2^2x_3^8\partial_1\partial_2+x_1^6x_3^{10}\partial_1\partial_2+x_1^8x_2^7x_3^2\partial_3+x_1^8x_2^5x_3^4\partial_3+x_1^4x_2^9x_3^4\partial_3+x_1^6x_2^5x_3^6\partial_3+x_1^4x_2^7x_3^6\partial_3+x_1^2x_2^9x_3^6\partial_3+$
$x_1^4x_2^5x_3^8\partial_3+x_1^2x_2^7x_3^8\partial_3+x_1^4x_2^3x_3^{10}\partial_3+x_1^5x_2^5x_3^2\partial_1^3\partial_2^2\partial_3+x_1^5x_2^3x_3^4\partial_1^3\partial_2^2\partial_3+x_1^2x_2^4x_3^6\partial_1^2\partial_2^3\partial_3+x_1^3x_2^6x_3^2\partial_1^3\partial_2^3\partial_3+$
$x_1^6x_2^5x_3^6+x_1^6x_2^3x_3^8+x_1^6x_2x_3^{10}+x_1^6x_2^4x_3^6\partial_1+x_1^6x_2^2x_3^8\partial_1+x_1^6x_3^{10}\partial_1+x_1^6x_2^5x_3^4\partial_1\partial_2+x_1^8x_2x_3^6\partial_1\partial_2+x_1^6x_2^8x_3^2\partial_3+$
$x_1^6x_2^6x_3^4\partial_3+x_1^4x_2^8x_3^4\partial_3+x_2^{10}x_3^6\partial_3+x_1^4x_2^4x_3^8\partial_3+x_2^8x_3^8\partial_3+x_1^4x_2^2x_3^{10}\partial_3+x_1^2x_2^6x_3^4\partial_1^2\partial_2^2\partial_3+x_1^4x_2^2x_3^6\partial_1^2\partial_2^2\partial_3+$
$x_1^2x_2^4x_3^6\partial_1^2\partial_2^2\partial_3+x_1^3x_2^2x_3^6\partial_1^3\partial_2^2\partial_3+x_1^2x_2^3x_3^6\partial_1^2\partial_2^3\partial_3+x_1^3x_2^5x_3^2\partial_1^3\partial_2^3\partial_3+x_1^6x_2^4x_3^6+x_1^6x_2^2x_3^8+x_1^6x_3^{10}+$
$x_1^6x_2^4x_3^4\partial_1\partial_2+x_1^8x_3^6\partial_1\partial_2+x_1^6x_2^7x_3^2\partial_3+x_1^6x_2^3x_3^6\partial_3+x_2^9x_3^6\partial_3+x_2^7x_3^8\partial_3+x_1^3x_2^5x_3^2\partial_1^3\partial_2^2\partial_3+x_1^3x_2^3x_3^4\partial_1^3\partial_2^2\partial_3+$
$x_1^2x_2^4x_3^4\partial_1^2\partial_2^3\partial_3+x_1^3x_2^6\partial_1^3\partial_2^3\partial_3+x_1^3x_2^4x_3^2\partial_1^3\partial_2^3\partial_3+x_1^6x_2^5x_3^4+x_1^8x_2x_3^6+x_2^3x_3^{12}+x_2x_3^{14}+x_1^6x_2^4x_3^4\partial_1+$
$x_1^8x_3^6\partial_1+x_1^8x_2^3x_3^2\partial_1\partial_2+x_1^6x_2x_3^6\partial_1\partial_2+x_1^6x_2^8\partial_3+x_1^8x_2^4x_3^2\partial_3+x_1^6x_2^6x_3^2\partial_3+x_1^6x_2^4x_3^4\partial_3+x_1^4x_2^4x_3^6\partial_3+$
$x_1^2x_2^6x_3^6\partial_3+x_1^2x_2^2x_3^6\partial_1^2\partial_2^2\partial_3+x_1^3x_2^2x_3^4\partial_1^3\partial_2^2\partial_3+x_1^2x_2^3x_3^4\partial_1^2\partial_2^3\partial_3+x_1^3x_2^5\partial_1^3\partial_2^3\partial_3+x_1^3x_2^3x_3^2\partial_1^3\partial_2^3\partial_3+$
$x_1^6x_2^4x_3^4+x_1^8x_3^6+x_2^2x_3^{12}+x_3^{14}+x_1^8x_2^2x_3^2\partial_1\partial_2+x_1^6x_3^6\partial_1\partial_2+x_1^6x_2^7\partial_3+x_1^8x_2^3x_3^2\partial_3+x_1^6x_2^5x_3^2\partial_3+x_1^4x_2^3x_3^6\partial_3+$
$x_1^2x_2^5x_3^6\partial_3+x_1^3x_2^5\partial_1^3\partial_2^2\partial_3+x_1^3x_2^3x_3^2\partial_1^3\partial_2^2\partial_3+x_1^8x_2^3x_3^2+x_1^6x_2x_3^6+x_1^2x_2^5x_3^6+x_1^2x_2^3x_3^8+x_1^8x_2^2x_3^2\partial_1+$
$x_1^6x_3^6\partial_1+x_1^6x_2^3x_3^2\partial_1\partial_2+x_1^6x_2x_3^4\partial_1\partial_2+x_1^6x_2^4x_3^2\partial_3+x_1^4x_2^2x_3^6\partial_3+x_2^6x_3^6\partial_3+x_1^2x_2^2x_3^4\partial_1^2\partial_2^2\partial_3+x_1^8x_2^2x_3^2+$
$x_1^6x_3^6+x_1^2x_2^4x_3^6+x_1^2x_2^2x_3^8+x_1^6x_2^2x_3^2\partial_1\partial_2+x_1^6x_3^4\partial_1\partial_2+x_1^6x_2^3x_3^2\partial_3+x_1^4x_2^5x_3^2\partial_3+x_2^5x_3^6\partial_3+x_1^6x_2^3x_3^2+$
$x_1^6x_2x_3^4+x_2^5x_3^6+x_2^3x_3^8+x_2x_3^{10}+x_1^6x_2^2x_3^2\partial_1+x_1^6x_3^4\partial_1+x_1^6x_2^4\partial_3+x_1^4x_2^4x_3^2\partial_3+x_1^6x_2^2x_3^2+x_1^6x_3^4+x_2^4x_3^6+$
$x_2^2x_3^8+x_3^{10}+x_1^6x_2^3\partial_3+x_1^4x_2^3x_3^2\partial_3+x_1^2x_2^5x_3^2\partial_3+x_1^2x_2^3x_3^4\partial_3+x_2^5x_3^4+x_1^2x_2x_3^6+x_1^2x_2^5\partial_1\partial_2+x_1^2x_2^3x_3^2\partial_1\partial_2+$
$x_1^4x_2^2x_3^2\partial_3+x_1^2x_2^4x_3^2\partial_3+x_1^2x_2^2x_3^4\partial_3+x_2^4x_3^4+x_1^2x_3^6+x_1^2x_2^4\partial_1\partial_2+x_1^2x_2^2x_3^2\partial_1\partial_2+x_1^4x_2^3\partial_3+x_1^4x_2x_3^2\partial_3+$
$x_2^5x_3^2\partial_3+x_2^3x_3^4\partial_3+x_1^2x_2^5+x_2x_3^6+x_1^2x_2^4\partial_1+x_1^2x_2^2x_3^2\partial_1+x_1^4x_2\partial_1\partial_2+x_2^5\partial_1\partial_2+x_2^3x_3^2\partial_1\partial_2+x_1^4x_2^2\partial_3+$
$x_1^4x_3^2\partial_3+x_2^4x_3^2\partial_3+x_2^2x_3^4\partial_3+x_1^2x_2^4+x_3^6+x_1^4\partial_1\partial_2+x_2^4\partial_1\partial_2+x_2^2x_3^2\partial_1\partial_2+x_1^2x_2x_3^2\partial_3+x_1^4x_2+x_2^5+$
$x_2x_3^4+x_1^4\partial_1+x_2^4\partial_1+x_2^2x_3^2\partial_1+x_1^2x_2\partial_1\partial_2+x_1^2x_3^2\partial_3+x_1^4+x_2^4+x_3^4+x_1^2\partial_1\partial_2+x_2x_3^2\partial_3+x_1^2x_2+$
$x_2^3+x_1^2\partial_1+x_2\partial_1\partial_2+x_3^2\partial_3+x_1^2+x_2^2+\partial_1\partial_2+x_2+\partial_1+1,$

$p_3=x_1^2x_2x_3\partial_1^9\partial_2^7\partial_3^9+x_1^2x_2\partial_1^9\partial_2^7\partial_3^8+x_1^2x_3\partial_1^9\partial_2^6\partial_3^9+x_1^2\partial_1^9\partial_2^6\partial_3^8+x_1^2x_2x_3\partial_1^7\partial_2^5\partial_3^9+x_1^2x_2\partial_1^7\partial_2^5\partial_3^8+$
$x_1^2x_3\partial_1^7\partial_2^4\partial_3^9+x_1x_2^4x_3^2\partial_1^8\partial_2^6+x_1^2\partial_1^7\partial_2^4\partial_3^8+x_1^2x_2x_3\partial_1^5\partial_2^3\partial_3^9+x_1\partial_1^7\partial_2^4\partial_3^9+x_1x_2^4x_3^2\partial_1^7\partial_2^5+x_1^2x_2\partial_1^5\partial_2^3\partial_3^8+$
$x_1^2x_3\partial_1^5\partial_2^2\partial_3^9+\partial_1^6\partial_2^4\partial_3^9+x_1^3\partial_1^6\partial_2^6\partial_3^3+x_1\partial_1^7\partial_2\partial_3^9+x_2^4x_3^2\partial_1^6\partial_2^5+x_1x_2^2\partial_1^8\partial_2^6+x_1^4x_2x_3\partial_1^3\partial_2^5\partial_3^3+$
$x_1^2\partial_1^5\partial_2^2\partial_3^8+x_1\partial_1^5\partial_2^2\partial_3^9+x_1^3\partial_1^7\partial_2^3\partial_3^3+\partial_1^6\partial_2\partial_3^9+x_1\partial_1^4\partial_2^2\partial_3^9+x_1x_2^6\partial_1^5\partial_2^3+x_1x_2^2\partial_1^7\partial_2^5+x_1\partial_1^8\partial_2^6+$
$x_1^4x_2\partial_1^3\partial_2^5\partial_3^2+x_1^4x_3\partial_1^3\partial_2^4\partial_3^3+\partial_1^4\partial_2^2\partial_3^9+x_1^6x_2^5x_3^2\partial_1+x_1^6x_2^3x_3^4\partial_1+x_1^2\partial_1^6\partial_2^3\partial_3^3+x_2^6\partial_1^4\partial_2^3+x_2^2\partial_1^6\partial_2^5+$
$x_1\partial_1^7\partial_2^5+x_1^4\partial_1^3\partial_2^4\partial_3^2+x_1^5x_2^5x_3^2+x_1^5x_2^3x_3^4+x_1^8x_2^3\partial_1+x_1^7x_2^4\partial_1+x_1^4x_2^7\partial_1+x_1^3x_2^8\partial_1+x_1^8x_2x_3^2\partial_1+x_1^3x_2^6x_3^2\partial_1+$
$x_1^4x_2^3x_3^4\partial_1+x_1^8x_2^2\partial_1\partial_2+x_1^4x_2^6\partial_1\partial_2+x_1^4x_2^4x_3^2\partial_1\partial_2+x_1\partial_1^6\partial_2^4\partial_3+x_1\partial_1^2\partial_3^9+x_1^8x_2^3+x_1^4x_2^7+x_1^4x_2^5x_3^2+$
$x_1x_2^6\partial_1^3\partial_2+x_1x_2^6\partial_1^2\partial_2^2+\partial_1^6\partial_2^5+x_1^2x_2x_3\partial_1^3\partial_2^3\partial_3+x_1^3\partial_1^3\partial_2^2\partial_3^3+x_1\partial_1\partial_3^9+x_1^7x_2^3+x_1^6x_2^4+x_1^3x_2^7+$
$x_1^2x_2^8+x_1^7x_2x_3^2+x_1^2x_2^6x_3^2+x_1^3x_2^3x_3^4+x_1^7x_2^2\partial_1+x_1^5x_2^4\partial_1+x_1^4x_2^5\partial_1+x_1^2x_2^7\partial_1+x_1x_2^8\partial_1+x_1^6x_2x_3^2\partial_1+$
$x_1^5x_2^2x_3^2\partial_1+x_1x_2^6x_3^2\partial_1+x_1^4x_2x_3^4\partial_1+x_1^2x_2^3x_3^4\partial_1+x_1^7x_2^2\partial_2+x_1^3x_2^6\partial_2+x_1^3x_2^4x_3^2\partial_2+x_1^4x_2^4\partial_1\partial_2+x_1^2x_2^6\partial_1\partial_2+$
$x_1^4x_2^2x_3^2\partial_1\partial_2+x_1^2x_2^4x_3^2\partial_1\partial_2+x_1\partial_1^7\partial_2\partial_3+x_1^4x_2^5+x_1^2x_2^7+x_1^4x_2^3x_3^2+x_1^2x_2^5x_3^2+x_1x_2^6\partial_1^2+x_1x_2^6\partial_1\partial_2+$

$x_2^6\partial_1^2\partial_2+x_1^2x_2\partial_1^3\partial_2^3+x_1^2x_3\partial_1^3\partial_2^2\partial_3+x_1^2\partial_1^2\partial_2^2\partial_3^3+\partial_3^9+x_1^6x_2^2+x_1^4x_2^4+x_1^3x_2^5+x_1x_2^7+x_2^8+x_1^5x_2x_3^2+$
$x_1^4x_2^2x_3^2+x_2^6x_3^2+x_1^3x_2x_3^4+x_1x_2^3x_3^4+x_1^7\partial_1+x_1^6x_2\partial_1+x_1^5x_2^2\partial_1+x_1^4x_2^3\partial_1+x_1^3x_2^4\partial_1+x_1^2x_2^5\partial_1+x_1^4x_2x_3^2\partial_1+$
$x_1^2x_2x_3^4\partial_1+x_1^3x_2^4\partial_2+x_1x_2^6\partial_2+x_1^3x_2^2x_3^2\partial_2+x_1x_2^4x_3^2\partial_2+x_1^6\partial_1\partial_2+x_1^4x_2^2\partial_1\partial_2+x_1^2x_2^4\partial_1\partial_2+x_1^2x_2^2x_3^2\partial_1\partial_2+$
$\partial_1^6\partial_2\partial_3+x_1^3\partial_2^2\partial_3^3+x_1^6x_2+x_1^4x_2^3+x_1^2x_2^5+x_1^2x_2^3x_3^2+x_1x_2^4x_3^2+x_2^6\partial_2+x_1^2\partial_1^3\partial_2^2+x_1^6+x_1^5x_2+x_1^4x_2^2+$
$x_1^3x_2^3+x_1^2x_2^4+x_1x_2^5+x_1^3x_2x_3^2+x_1x_2x_3^4+x_1^2x_2^3\partial_1+x_1x_2^4\partial_1+x_1^3x_3^2\partial_1+x_1^2x_2x_3^2\partial_1+x_1^5\partial_2+x_1^3x_2^2\partial_2+$
$x_1x_2^4\partial_2+x_1x_2^2x_3^2\partial_2+x_1^2x_2^2\partial_1\partial_2+x_1^2x_2^3+x_1\partial_1^3\partial_3+x_1x_2^3+x_2^4+x_1^2x_3^2+x_1x_2x_3^2+x_1^3\partial_1+x_1x_3^2\partial_1+$
$x_1x_2^2\partial_2+x_1x_2^2+\partial_1^2\partial_3+x_1^2+x_3^2+x_1\partial_1+x_1\partial_3+x_1+1.$

(4) **Example 6.4.4**

The polynomials $p_1,p_2\in A_3=\mathbb{F}_3[x_1,x_2,x_3,\partial_1,\partial_2,\partial_3]$ of the public key Q are:

$p_1=x_3^{31}\partial_1^7\partial_2^7-x_3^{31}\partial_1^7\partial_2^4-x_3^{19}\partial_1^{16}\partial_2^7+x_3^7\partial_1^{25}\partial_2^{10}+x_3\partial_1^{25}\partial_2^{10}\partial_3^6-x_3^{30}\partial_1^5\partial_2\partial_3^5-x_1x_3^{18}\partial_1^{13}\partial_2^4\partial_3^5-$
$x_3^{18}\partial_1^{14}\partial_2^4\partial_3^5-x_3^7\partial_1^{22}\partial_2^{10}+x_3^{19}\partial_1^{12}\partial_2^3\partial_3^5-x_3\partial_1^{22}\partial_2^{10}\partial_3^6-x_3^{16}\partial_1^7\partial_2^7\partial_3^9+x_3^4\partial_1^{19}\partial_2^7\partial_3^9+x_3^{30}\partial_1^5\partial_2\partial_3^2-$
$x_1x_3^{15}\partial_1^{16}\partial_2^4\partial_3^2-x_3^{15}\partial_1^{17}\partial_2^4\partial_3^2-x_3^{30}\partial_1^2\partial_2\partial_3^5-x_3^{31}\partial_1^4\partial_2-x_1^3x_3^{16}\partial_1^{10}\partial_2^7-x_3^{19}\partial_1^{10}\partial_2^7+x_3^{16}\partial_1^{13}\partial_2^7-$
$x_3^7\partial_1^{22}\partial_2^7-x_1^3x_3^4\partial_1^{19}\partial_2^{10}+x_3^4\partial_1^{22}\partial_2^{10}+x_3^{16}\partial_1^{15}\partial_2^3\partial_3^2+x_3^3\partial_1^{18}\partial_2^{12}\partial_3^3+x_3\partial_1^{19}\partial_2^{10}\partial_3^6+x_3^{30}\partial_1^2\partial_2\partial_3^2+$
$x_1x_3^{18}\partial_1^{10}\partial_2\partial_3^5-x_3^{18}\partial_1^{11}\partial_2\partial_3^5+x_1x_3^{15}\partial_1^{10}\partial_2^4\partial_3^5+x_3^{15}\partial_1^{11}\partial_2^4\partial_3^5+x_1x_3^6\partial_1^{19}\partial_2^4\partial_3^5+x_1\partial_1^{19}\partial_2^4\partial_3^{11}-$
$x_3^{31}\partial_1\partial_2-x_1^3x_3^{16}\partial_1^7\partial_2^7-x_3^{16}\partial_1^{10}\partial_2^7-x_3^7\partial_1^{19}\partial_2^7-x_3^4\partial_1^{19}\partial_2^{10}+\partial_1^{21}\partial_2^{12}+x_3^{15}\partial_1^6\partial_2^9\partial_3^3-x_3^{19}\partial_1^9\partial_3^5-$
$x_3^{16}\partial_1^9\partial_2^3\partial_3^5-x_3^7\partial_1^{18}\partial_2^3\partial_3^5-x_3\partial_1^{19}\partial_2^7\partial_3^6-x_3\partial_1^{16}\partial_2^{10}\partial_3^6-x_3\partial_1^{16}\partial_2^7\partial_3^9-x_3\partial_1^{18}\partial_2^3\partial_3^{11}-x_3^{18}\partial_1^{11}\partial_2\partial_3^2-$
$x_1x_3^{15}\partial_1^{13}\partial_2\partial_3^2-x_3^{15}\partial_1^{14}\partial_2\partial_3^2-x_1x_3^{15}\partial_1^{10}\partial_2^4\partial_3^2-x_3^{15}\partial_1^{11}\partial_2^4\partial_3^2+x_3^6\partial_1^{20}\partial_2^4\partial_3^2-x_1x_3^6\partial_1^{16}\partial_2^4\partial_3^5+$
$\partial_1^{20}\partial_2^4\partial_3^8-x_1\partial_1^{16}\partial_2^4\partial_3^{11}-x_1x_3^{15}\partial_1\partial_2\partial_3^{14}+x_1x_3^3\partial_1^{13}\partial_2\partial_3^{14}-x_1^3x_3^4\partial_1^{16}\partial_2^7-x_3^4\partial_1^{19}\partial_2^7+x_1^3x_3\partial_1^{16}\partial_2^{10}+$
$x_3\partial_1^{19}\partial_2^{10}+x_3^{16}\partial_1^{12}\partial_3^2+x_3^{16}\partial_1^9\partial_2^3\partial_3^2-x_3^3\partial_1^{15}\partial_2^9\partial_3^3-\partial_1^{15}\partial_2^{12}\partial_3^3+x_3^7\partial_1^{15}\partial_2^3\partial_3^5+x_3\partial_1^{15}\partial_2^3\partial_3^{11}+$
$x_3^{16}\partial_3^{14}-x_3^4\partial_1^{12}\partial_3^{14}-x_3^6\partial_1^{17}\partial_2^4\partial_3^2-x_1x_3^{18}\partial_1^4\partial_2\partial_3^5+x_1^3x_3^{15}\partial_1^5\partial_2\partial_3^5-x_1x_3^{15}\partial_1^7\partial_2\partial_3^5+x_3^{15}\partial_1^8\partial_2\partial_3^5-$
$x_1x_3^6\partial_1^{16}\partial_2\partial_3^5-x_1^4x_3^3\partial_1^{13}\partial_2^4\partial_3^5+x_1x_3^3\partial_1^{16}\partial_2^4\partial_3^5-\partial_1^{17}\partial_2^4\partial_3^8-x_3^{15}\partial_1^2\partial_2\partial_3^{11}+x_3^3\partial_1^{14}\partial_2\partial_3^{11}+x_1\partial_1^{13}\partial_2^4\partial_3^{11}+$
$x_3^4\partial_1^{16}\partial_2^7-\partial_1^{18}\partial_2^9-x_3\partial_1^{16}\partial_2^{10}+\partial_1^{15}\partial_2^{12}+x_3^{15}\partial_1^3\partial_2^6\partial_3^3+x_3^{19}\partial_1^3\partial_3^5+x_3^{16}\partial_1^6\partial_3^5+x_3^7\partial_1^{15}\partial_3^5+x_1^3x_3^4\partial_1^{12}\partial_2^3\partial_3^5-$
$x_3^4\partial_1^{15}\partial_2^3\partial_3^5-x_3\partial_1^{13}\partial_2^7\partial_3^6+x_1^3x_3\partial_1^7\partial_2^7\partial_3^9-x_3\partial_1^{12}\partial_2^3\partial_3^{11}-x_1^3x_3^{15}\partial_1^5\partial_2\partial_3^2-x_3^{18}\partial_1^5\partial_2\partial_3^2-x_1x_3^{15}\partial_1^7\partial_2\partial_3^2-$
$x_3^6\partial_1^{17}\partial_2\partial_3^2-x_1^3x_3^3\partial_1^{14}\partial_2^4\partial_3^2+x_3^3\partial_1^{17}\partial_2^4\partial_3^2+x_1^3x_3^{15}\partial_1^2\partial_2\partial_3^5+x_1x_3^{15}\partial_1^4\partial_2\partial_3^5-x_3^{15}\partial_1^5\partial_2\partial_3^5-x_1x_3^6\partial_1^{13}\partial_2\partial_3^5-$
$x_1x_3^3\partial_1^{13}\partial_2^4\partial_3^5+\partial_1^{14}\partial_2^4\partial_3^8-x_1\partial_1^{13}\partial_2\partial_3^{11}-x_1\partial_1^{10}\partial_2^4\partial_3^{11}-x_1\partial_1^{10}\partial_2\partial_3^{14}+x_1^3x_3\partial_1^{13}\partial_2^7-x_3\partial_1^{16}\partial_2^7+$
$x_3^{16}\partial_1^6\partial_3^2-x_3^{15}\partial_1^3\partial_2^3\partial_3^3+x_3^3\partial_1^{12}\partial_2^6\partial_3^3-x_1^3\partial_1^9\partial_2^9\partial_3^3+\partial_1^{12}\partial_2^9\partial_3^3-x_3^{16}\partial_1^3\partial_3^5+x_3^7\partial_1^{12}\partial_3^5+x_3^4\partial_1^{12}\partial_2^3\partial_3^5-$
$x_3\partial_1^7\partial_2^7\partial_3^9+x_3\partial_1^{12}\partial_3^{11}+x_3\partial_1^9\partial_2^3\partial_3^{11}+x_3\partial_1^9\partial_3^{14}-x_1^3x_3^{15}\partial_1^2\partial_2\partial_3^2-x_3^{15}\partial_1^5\partial_2\partial_3^2-x_3^6\partial_1^{14}\partial_2\partial_3^2-$
$x_3^3\partial_1^{14}\partial_2^4\partial_3^2-x_1x_3^{15}\partial_1\partial_2\partial_3^5-x_3^{15}\partial_1^2\partial_2\partial_3^5-x_1^4x_3^3\partial_1^{10}\partial_2\partial_3^5-x_1x_3^3\partial_1^{13}\partial_2\partial_3^5+x_1^4\partial_1^{10}\partial_2^4\partial_3^5+x_1\partial_1^{13}\partial_2^4\partial_3^5-$
$\partial_1^{14}\partial_2\partial_3^8-\partial_1^{11}\partial_2^4\partial_3^8-\partial_1^{11}\partial_2\partial_3^{11}+\partial_1^{15}\partial_2^6-x_1^3x_3\partial_1^{10}\partial_2^7-x_3^4\partial_1^{10}\partial_2^7+\partial_1^{12}\partial_2^9-x_3^{15}\partial_2^3\partial_3^3-x_1^3\partial_1^6\partial_2^9\partial_3^3+$
$\partial_1^9\partial_2^9\partial_3^3+x_3^{16}\partial_3^5+x_1^3x_3^4\partial_1^9\partial_3^5+x_3^4\partial_1^{12}\partial_3^5-x_1^3x_3\partial_1^9\partial_2^3\partial_3^5-x_3\partial_1^{12}\partial_2^3\partial_3^5-x_1^3x_3^3\partial_1^{11}\partial_2\partial_3^2-x_3^3\partial_1^{14}\partial_2\partial_3^2+$
$x_1^3\partial_1^{11}\partial_2^4\partial_3^2+\partial_1^{14}\partial_2^4\partial_3^2+x_1x_3^3\partial_1^{10}\partial_2\partial_3^5-x_1\partial_1^{10}\partial_2^4\partial_3^5-x_1\partial_1^7\partial_2\partial_3^{11}+x_1^4\partial_1\partial_2\partial_3^{14}-x_1^3x_3\partial_1^7\partial_2^7+$
$x_3\partial_1^{10}\partial_2^7-x_1^3\partial_1^6\partial_2^6\partial_3^3-\partial_1^9\partial_2^6\partial_3^3+\partial_1^6\partial_2^9\partial_3^3-x_3^4\partial_1^9\partial_3^5+x_3\partial_1^9\partial_2^3\partial_3^5+x_3\partial_1^6\partial_3^{11}-x_1^3x_3\partial_3^{14}+x_3^3\partial_1^{11}\partial_2\partial_3^2-$

$\partial_1^{11}\partial_2^4\partial_3^2 + x_1^4\partial_1^7\partial_2\partial_3^5 - x_1\partial_1^{10}\partial_2\partial_3^5 - \partial_1^8\partial_2\partial_3^8 + x_1^3\partial_1^2\partial_2\partial_3^{11} - x_1\partial_1\partial_2\partial_3^{14} + \partial_1^9\partial_2^6 - x_3\partial_1^7\partial_2^7 -$
$x_1^3\partial_1^3\partial_2^6\partial_3^3 + \partial_1^6\partial_2^6\partial_3^3 - x_1^3x_3\partial_1^6\partial_3^5 + x_3\partial_1^9\partial_3^5 + x_3\partial_3^{14} + x_1^3\partial_1^8\partial_2\partial_3^2 - \partial_1^{11}\partial_2\partial_3^2 - x_1^4\partial_1^4\partial_2\partial_3^5 - x_1x_3^3\partial_1^4\partial_2\partial_3^5 -$
$\partial_1^2\partial_2\partial_3^{11} + \partial_1^3\partial_2^6\partial_3^3 + x_1^3x_3\partial_1^3\partial_3^5 + x_3^4\partial_1^3\partial_3^5 - x_1^3\partial_1^5\partial_2\partial_3^2 - x_3^3\partial_1^5\partial_2\partial_3^2 - x_1^4\partial_1\partial_2\partial_3^5 + x_1\partial_1^4\partial_2\partial_3^5 +$
$x_1^3x_3\partial_3^5 - x_3\partial_1^3\partial_3^5 - x_1^3\partial_1^2\partial_2\partial_3^2 + \partial_1^5\partial_2\partial_3^2 - x_1\partial_1\partial_2\partial_3^5 - \partial_1^3\partial_2^3 + x_3\partial_3^5 - \partial_1^2\partial_2\partial_3^2 + \partial_1^3 + 1,$
$p_2 = x_1^2x_3^4\partial_1^{20}\partial_2^{12} + x_1^2x_3^{16}\partial_1^8\partial_2^9 + x_3^4\partial_1^{18}\partial_2^{12} + x_1^2x_3^4\partial_1^{17}\partial_2^9 - x_1^2x_3\partial_1^{17}\partial_2^{12} + x_3^{16}\partial_1^6\partial_2^9 - x_1^{10}x_2^3x_3\partial_1^{11}\partial_2^4 +$
$x_1^{10}x_2^3x_3\partial_1^{10}\partial_2^3\partial_3^2 + x_1^2x_3\partial_1^{11}\partial_2^6\partial_3^9 + x_1x_3\partial_1^{11}\partial_2^4\partial_3^{12} - x_1x_3\partial_1^{10}\partial_2^3\partial_3^{14} + x_3^4\partial_1^{15}\partial_2^9 - x_3\partial_1^{15}\partial_2^{12} - x_1x_3^{16}\partial_1^7\partial_2^3 +$
$x_1^9x_2^3x_3\partial_1^{10}\partial_2^4 - x_1x_3^4\partial_1^{16}\partial_2^6 - x_1^{10}x_2^3\partial_1^{10}\partial_2^3\partial_3 + x_1^9x_2^3x_3\partial_1^9\partial_2^3\partial_3^2 - x_3\partial_1^{10}\partial_2^4\partial_3^{12} + x_1\partial_1^{10}\partial_2^3\partial_3^{13} - x_3\partial_1^9\partial_2^3\partial_3^{14} +$
$x_1x_2^6x_3\partial_1^{14}\partial_2^4 + x_1x_2^3x_3^4\partial_1^{14}\partial_2^4 - x_1^2x_3\partial_1^{14}\partial_2^9 - x_1x_2^6x_3\partial_1^{13}\partial_2^3\partial_3^2 - x_1x_2^3x_3^4\partial_1^{13}\partial_2^3\partial_3^2 - x_1x_3^{16}\partial_1^5\partial_2\partial_3^3 -$
$x_1x_3^4\partial_1^{14}\partial_2^4\partial_3^3 + x_1x_3^{16}\partial_1^4\partial_3^5 + x_1x_3^4\partial_1^{13}\partial_2^3\partial_3^5 + x_1^2x_3\partial_1^8\partial_2^6\partial_3^9 - x_3^{16}\partial_1^6\partial_2^3 - x_3^4\partial_1^{15}\partial_2^6 - x_1x_3^{16}\partial_1^4\partial_2^3\partial_3 -$
$x_1^9x_2^3\partial_1^9\partial_2^3\partial_3 - x_1x_3^4\partial_1^{13}\partial_2^6\partial_3 - x_1x_3^{16}\partial_1\partial_2^3\partial_3^4 + x_3\partial_1^9\partial_2^6\partial_3^9 + \partial_1^9\partial_2^3\partial_3^{13} - x_1x_3^{16}\partial_1^4\partial_2^3 - x_2^6x_3\partial_1^{13}\partial_2^4 -$
$x_2^3x_3^4\partial_1^{13}\partial_2^4 + x_1x_2^6\partial_1^{13}\partial_2^3\partial_3 + x_1x_2^3x_3^3\partial_1^{13}\partial_2^3\partial_3 - x_2^6x_3\partial_1^{12}\partial_2^3\partial_3^2 - x_2^3x_3^4\partial_1^{12}\partial_2^3\partial_3^2 + x_3^{16}\partial_1^4\partial_2\partial_3^3 +$
$x_3^4\partial_1^{13}\partial_2^4\partial_3^3 - x_1x_3^{15}\partial_1^4\partial_3^4 - x_1x_3^3\partial_1^{13}\partial_2^3\partial_3^4 + x_3^{16}\partial_1^3\partial_3^5 + x_3^4\partial_1^{12}\partial_2^3\partial_3^5 + x_1x_3\partial_1^{10}\partial_2^3\partial_3^9 - x_1^2x_3^{16}\partial_1^2\partial_2^3 -$
$x_1x_3^{15}\partial_1^4\partial_2^3 - x_1^7x_3\partial_1^{11}\partial_2^4 + x_1x_2^6x_3\partial_1^{11}\partial_2^4 + x_1x_2^3x_3^4\partial_1^{11}\partial_2^4 - x_1x_3^3\partial_1^{13}\partial_2^6 - x_3^{16}\partial_1^3\partial_2^3\partial_3 - x_3^4\partial_1^{12}\partial_2^6\partial_3 +$
$x_1^7x_3\partial_1^{10}\partial_2^3\partial_3^2 - x_1x_2^6x_3\partial_1^{10}\partial_2^3\partial_3^2 - x_1x_2^3x_3^4\partial_1^{10}\partial_2^3\partial_3^2 - x_1x_3^{16}\partial_1^2\partial_2\partial_3^3 - x_1x_3^{15}\partial_1\partial_2^3\partial_3^3 - x_3^{16}\partial_2^3\partial_3^4 +$
$x_1x_3^{16}\partial_1\partial_3^5 + x_1^2x_3\partial_1^8\partial_2^3\partial_3^9 + x_1x_3\partial_1^8\partial_2\partial_3^{12} - x_1x_3\partial_1^7\partial_3^{14} - x_3^{16}\partial_1^3\partial_2^3 - x_3\partial_1^{12}\partial_2^9 - x_1x_3^{16}\partial_1\partial_2^3\partial_3 +$
$x_2^6\partial_1^{12}\partial_2^3\partial_3 + x_2^3x_3^3\partial_1^{12}\partial_2^3\partial_3 - x_3^{15}\partial_1^3\partial_3^4 - x_3^3\partial_1^{12}\partial_2^3\partial_3^4 + x_3\partial_1^9\partial_2^3\partial_3^9 + x_3\partial_1^6\partial_2^6\partial_3^9 - x_1x_3^{16}\partial_1^4 - x_3^{15}\partial_1^3\partial_2^3 +$
$x_1x_3^4\partial_1^{13}\partial_2^3 + x_1^6x_3\partial_1^{10}\partial_2^4 - x_2^6x_3\partial_1^{10}\partial_2^4 - x_2^3x_3^4\partial_1^{10}\partial_2^4 - x_3^3\partial_1^{12}\partial_2^6 + x_1x_3\partial_1^{13}\partial_2^6 - x_1^7\partial_1^{10}\partial_2^3\partial_3 + x_1x_2^6\partial_1^{10}\partial_2^3\partial_3 +$
$x_1x_2^3x_3^3\partial_1^{10}\partial_2^3\partial_3 + x_1^6x_3\partial_1^9\partial_2^3\partial_3^2 - x_2^6x_3\partial_1^9\partial_2^3\partial_3^2 - x_2^3x_3^4\partial_1^9\partial_2^3\partial_3^2 + x_3^{16}\partial_1\partial_2\partial_3^3 - x_3^{15}\partial_2^3\partial_3^3 - x_1x_3^{15}\partial_1\partial_3^4 +$
$x_3^{16}\partial_3^5 - x_3\partial_1^7\partial_2\partial_3^{12} + x_1\partial_1^7\partial_3^{13} - x_3\partial_1^6\partial_3^{14} - x_1x_3^{15}\partial_1\partial_2^3 - x_1^2x_3^4\partial_1^{11}\partial_2^3 - x_1x_3\partial_1^{14}\partial_2^4 + x_1^2x_3\partial_1^8\partial_2^9 -$
$x_3^{16}\partial_2^3\partial_3 + x_1x_3\partial_1^{13}\partial_2^3\partial_3^2 - x_1x_3^4\partial_1^{11}\partial_2\partial_3^3 + x_1x_3\partial_1^{11}\partial_2^4\partial_3^3 + x_1x_3^4\partial_1^{10}\partial_3^5 - x_1x_3\partial_1^{10}\partial_2^3\partial_3^5 - x_3^{16}\partial_1^3 -$
$x_3^{16}\partial_2^3 + x_3^4\partial_1^{12}\partial_2^3 + x_3\partial_1^{12}\partial_2^6 - x_1^6\partial_1^9\partial_2^3\partial_3 + x_2^6\partial_1^9\partial_2^3\partial_3 + x_2^3x_3^3\partial_1^9\partial_2^3\partial_3 - x_1x_3^4\partial_1^{10}\partial_2^3\partial_3 + x_1x_3\partial_1^{10}\partial_2^6\partial_3 -$
$x_3^{15}\partial_3^4 + x_3\partial_1^6\partial_2^3\partial_3^9 + \partial_1^6\partial_3^{13} - x_1x_3^{16}\partial_1 - x_3^{15}\partial_2^3 + x_3\partial_1^{13}\partial_2^4 - x_1\partial_1^{13}\partial_2^3\partial_3 + x_3\partial_1^{12}\partial_2^3\partial_3^2 + x_3^4\partial_1^{10}\partial_2\partial_3^3 -$
$x_3\partial_1^{10}\partial_2^4\partial_3^3 - x_1x_3^3\partial_1^{10}\partial_3^4 + x_1\partial_1^{10}\partial_2^3\partial_3^4 + x_3^4\partial_1^9\partial_3^5 - x_3\partial_1^9\partial_2^3\partial_3^5 - x_1x_3^3\partial_1^{10}\partial_2^3 - x_1x_3\partial_1^{11}\partial_2^4 + x_1\partial_1^{10}\partial_2^6 -$
$x_3^4\partial_1^9\partial_2^3\partial_3 + x_3\partial_1^9\partial_2^6\partial_3 + x_1x_3\partial_1^{10}\partial_2^3\partial_3^2 + x_1^2x_3\partial_1^5\partial_3^9 - x_3^{16} - x_3^4\partial_1^9\partial_2^3 + x_3\partial_1^6\partial_2^9 + x_1^{10}x_2^3x_3\partial_1\partial_3 -$
$\partial_1^{12}\partial_2^3\partial_3 - x_3^3\partial_1^9\partial_3^4 + \partial_1^9\partial_2^3\partial_3^4 - x_1x_3\partial_1^4\partial_3^{10} - x_1x_3\partial_1\partial_3^{13} - x_1x_3^4\partial_1^{10} - x_3^3\partial_1^9\partial_2^3 - x_1x_3\partial_1^{10}\partial_2^3 +$
$x_3\partial_1^{10}\partial_2^4 + \partial_1^9\partial_2^6 - x_1\partial_1^{10}\partial_2^3\partial_3 + x_3\partial_1^9\partial_2^3\partial_3^2 - x_1x_3\partial_1^4\partial_3^9 + x_1^{10}x_2^3\partial_1 - x_1^2x_3^4\partial_1^8 + x_1^2x_3\partial_1^8\partial_2^3 + x_1^9x_2^3x_3\partial_3 +$
$x_1x_3\partial_1^8\partial_2\partial_3^3 - x_1x_3\partial_1^7\partial_3^5 - x_1\partial_1^4\partial_3^9 - x_3\partial_1^3\partial_3^{10} - x_1\partial_1\partial_3^{12} - x_3\partial_3^{13} - x_3^4\partial_1^9 - x_3\partial_1^9\partial_2^3 - x_1x_2^6x_3\partial_1^4\partial_3 -$
$x_1x_2^3x_3^4\partial_1^4\partial_3 + x_1x_3\partial_1^7\partial_2^3\partial_3 - \partial_1^9\partial_2^3\partial_3 + x_1x_3^4\partial_1^4\partial_3^4 - x_1x_3\partial_1\partial_3^{10} + x_1^9x_2^3 - x_1x_3\partial_1^7\partial_2^3 - x_3\partial_1^7\partial_2\partial_3^3 +$
$x_1\partial_1^7\partial_3^4 - x_3\partial_1^6\partial_3^5 - \partial_1^3\partial_3^9 - \partial_3^{12} - x_1x_2^6\partial_1^4 - x_1x_2^3x_3^3\partial_1^4 + x_1\partial_1^7\partial_2^3 - x_2^6x_3\partial_1^3\partial_3 - x_2^3x_3^4\partial_1^3\partial_3 + x_3\partial_1^6\partial_2^3\partial_3 +$
$x_1x_3^3\partial_1^4\partial_3^3 - x_1x_3\partial_1^5\partial_2\partial_3^3 + x_3^4\partial_1^3\partial_3^4 + x_1x_3\partial_1^4\partial_3^5 - x_1\partial_1\partial_3^9 - x_3\partial_3^{10} - x_3^4\partial_1^6 + x_1^7x_3\partial_1\partial_3 - x_1x_2^6x_3\partial_1\partial_3 -$
$x_1x_2^3x_3^4\partial_1\partial_3 - x_1x_3\partial_1^4\partial_2^3\partial_3 + \partial_1^6\partial_3^4 - x_1x_3\partial_1\partial_2^3\partial_3^4 - x_2^6\partial_1^3 - x_2^3x_3^3\partial_1^3 + x_1x_3\partial_1^7 - x_1x_3\partial_1^4\partial_2^3 + \partial_1^6\partial_2^3 +$
$x_3^3\partial_1^3\partial_3^3 + x_3\partial_1^4\partial_2\partial_3^3 - x_1\partial_1^4\partial_3^4 + x_3\partial_1^3\partial_3^5 - \partial_3^9 + x_1^7\partial_1 - x_1x_2^6\partial_1 - x_1x_2^3x_3^3\partial_1 + x_1^2x_3\partial_1^5 - x_1^2x_3\partial_1^2\partial_2^3 -$

$$x_1\partial_1^4\partial_2^3+x_1^6x_3\partial_3-x_2^6x_3\partial_3-x_2^3x_3^4\partial_3-x_3\partial_1^3\partial_2^3\partial_3-x_1x_3\partial_1^2\partial_2\partial_3^3-x_1\partial_1\partial_2^3\partial_3^3-x_3\partial_2^3\partial_3^4+x_1x_3\partial_1\partial_3^5+$$
$$x_3\partial_1^6-x_3\partial_1^3\partial_2^3+x_1x_3\partial_1^4\partial_3-x_1x_3\partial_1\partial_2^3\partial_3-x_1x_3\partial_1\partial_3^4-\partial_1^3\partial_3^4+x_1^6-x_2^6-x_2^3x_3^3-x_1x_3\partial_1^4-$$
$$\partial_1^3\partial_2^3+x_3\partial_1\partial_2\partial_3^3-\partial_2^3\partial_3^3-x_1\partial_1\partial_3^4+x_3\partial_3^5+x_1\partial_1^4-x_1\partial_1\partial_2^3+x_3\partial_1^3\partial_3-x_3\partial_2^3\partial_3-x_1\partial_1\partial_3^3-$$
$$x_3\partial_3^4-x_3\partial_2^3+x_1x_3\partial_1\partial_3-\partial_3^4-x_1x_3\partial_1+\partial_1^3-\partial_2^3-\partial_3^3+x_1\partial_1+x_3\partial_3-x_3+1.$$

Bibliography

[1] P. Ackermann and M. Kreuzer. Gröbner basis cryptosystems. *Applicable Alg. in Eng., Commun. and Comput.*, 17:173–194, 2006.

[2] R. Ali and M. Kreuzer. Weyl Gröbner basis cryptosystems, preprint 2011 (submitted).

[3] I. Anshel, M. Anshel, and D. Goldfeld. An algebraic method for public-key cryptography. *Math. Res. Lett.*, 6:287–291, 1999.

[4] ApCoCoA team. ApCoCoA: Applied Computations in Commutative Algebra. Available at `http://www.apcocoa.org`.

[5] M. Aschenbrenner and A. Leykin. Degree bound for gröbner bases in algebras of solvable type. *Journal of Pure and Applied Algebra*, 213.

[6] F. Bao, R. H. Deng, W. Geiselmann, C. Schnorr, R. Steinwandt, and H. Wu. Cryptoanalysis of two sparse polynomial based public key cryptosystems. In *Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography*, PKC '01, pages 153–164, London, UK, 2001. Springer-Verlag.

[7] A. Belov-Kanel and M. Kontsevich. Automorphisms of the Weyl algebra. Available at `arxiv:math/0512169`, 2008.

[8] Boo Barkee *et al.* Why you cannot even hope to use Gröbner bases in public key cryptology. *J. Symb. Comput.*, 18:497–501, 1994.

[9] J. Buchmann. *Introduction to Cryptography*. Springer, New York, 2001.

[10] S. Bulygin. Chosen-ciphertext attack on noncommutative polly-cracker. Available at `http://arxiv.org/abs/cs/0508015v2`.

[11] CoCoA team. CoCoA: a system for doing Computations in Commutative Algebra. Available at `http://cocoa.dima.unige.it`.

[12] S. Coutinho. *A Primer of Algebraic D-Modules*. Cambridge University Press, Cambridge, 1995.

[13] A. Dickenstein, N. Fitchas, M. Giusti, and C. Sessa. The membership problem for unmixed polynomial ideals is solvable in single exponential time. *Discrete Applied Mathematics*, 33:73–94, 1991.

[14] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22:644–654, 1976.

[15] J. Ding, J. E. Gower, and D. S. Schmidt. *Multivariate Public Key Cryptosystems*. Springer Verlag, New York, 2006.

[16] Dumas, Gautier, Giesbrecht, Giorgi, Hovinen, Kaltofen, Saunders, Turner, and Villard. Linbox: A generic library for exact linear algebra. In A. Cohen, X-S Gao, and N. Takayama, editors, *Mathematical Software: ICMS 2002 (Proceedings of the first International Congress of Mathematical Software)*, pages 40–50. World Scientific, 2002.

[17] Taher ElGamas. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. on Information Theory*, IT-31(4):469–472, 1985.

[18] M. Fellows and N. Koblitz. Combinatorial cryptosystems galore! *Contemp. Math.*, 168:51–61, 1994.

[19] K. R. Goodearl and R. B. Warfield. *An Introduction to Noncommutative Noetherian Rings*. Volume 16 of London Mathematical Society Student Texts. Cambridge University Press, Cambridge, 1989.

[20] D. Grant, K. Krastev, D. Lieman, and I. Shparlinski. A public key cryptosystem based on sparse polynomials. In *International Conference on Coding Theory, Cryptography and Related Areas, Proceedings of ICCC 1998, Johannes*

Buchmann, PKC '01, pages 114–121, 2000.

[21] D. R. Grayson and M. E. Stillman. Macaulay 2, a software system for research in algebraic geometry. Available at `http://www.math.uiuc.edu/Macaulay2`.

[22] G.-M. Greuel, G. Pfister, and H. Schönemann. Singular 3.0. A computer algebra system for polynomial computations. Available at `http://www.singular.uni-kl.de`.

[23] D. Hofheinz and R. Steinwandt. A "differential" attack on Polly Cracker. *Int. J. Inf. Secur.*, 1, 2002.

[24] A. Kandri-Rody and V. Weispfenning. Non-commutative Gröbner bases in algebras of solvable type. *J. Symb. Comput.*, 9:1–26, 1990.

[25] N. Koblitz. *Algebraic Aspects of Cryptography*, volume 3 of *Alg. and Comput. in Math.* Springer Verlag, Berlin, 1998.

[26] H. Kredel. *Solvable Polynomial Rings*. Verlag Shaker, Aachen, 1993.

[27] M. Kreuzer and L. Robbiano. *Computational Commutative Algebra 1*. Springer Verlag, Heidelberg, 2000.

[28] M. Kreuzer and L. Robbiano. *Computational Commutative Algebra 2*. Springer Verlag, Heidelberg, 2005.

[29] A. V. Lakeyev and V. Kreinovich. Np-hard classes of linear algebraic systems with uncertainties. *Reliable Computing*, 3:51–81, 1997.

[30] V. Levandovskyy. Non-commutative computer algebra for polynomial algebras: Gröbner bases, applications and implementation. Dissertation, Universität Kaiserslautern, 2005.

[31] V. Levandovskyy. *Plural, a noncommutative extension of Singular: past, present and future*, volume 36 of *Reports on Computer Algebra*. Universität Kaiserslautern, 2006.

[32] A. Leykin. D-modules for Macaulay 2. In ???, editor, *Mathematical Software, Beijing 2002*, pages 169–179, River Edge, 2002. World Sci. Publishing.

[33] H. Li. *Noncommutative Gröbner Basis and Filtered-Graded Transfer*.

Springer Verlag.

[34] L. Ly. Polly two – a new algebraic polynomial-based public-key scheme. *Applicable Alg. in Eng., Commun. and Comput.*, 17:267–283, 2006.

[35] L. Van Ly. Polly two; a public key cryptosystem based on polly-cracker. Ph.D. Dissertation, Rhur Universität Bochum, Germany, 2002.

[36] Ernst W. Mayr and Albert R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Advances in Mathematics*, 46(3):305 – 329, 1982.

[37] J. C. McConnel and J. C. Robson. *Noncommutative Noetherian Rings*. Pure and Applied Mathematics. John Wiley and Sons, Chichester, 1987.

[38] Alfred J. Menezes, Paul C. Van Oorschot, Scott A. Vanstone, and R. L. Rivest. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1997.

[39] T. Mora. An introduction to commutative and non-commutative gröbner basis. *Theor. Comp. Sci.*, 134:131–173, 1994.

[40] M. Noro, T. Shimoyama, and T. Takeshima. Risa/Asir, a computer algebra system. Available at `ftp://archives.cs.ehime-u.ac.jp/pub/asir2000`.

[41] T. Rai. Infinite Gröbner bases and noncommutative Polly Cracker cryptosystems. Dissertation. Virginia Polytechnic Institute, Blacksburg, 2004.

[42] T. Rai and S. Bulygin. Noncommutative Polly Cracker-type cryptosystems and chosen-ciphertext security. Cryptology ePrint Archive, Report 2008/504. Available at `http://eprint.iacr.org`.

[43] P. Revoy. Algèbres de Weyl en charactéristique p. *Compt. Rend. Acad. Sci. Paris, Ser. A*, 276:225–228.

[44] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[45] M. Saito, B. Sturmfels, and N. Takayama. *Gröbner Deformations of Hyperge-*

ometric Differential Equations. Algorithms and Computation in Mathematics 6. Springer Verlag, Berlin, 2000.

[46] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.*, 41:303–332, June 1999.

[47] R. Steinwandt. A ciphertext-only attack on polly two. *Applicable Algebra Engineering, Communication and Computing*, 21(2).

[48] R. Steinwandt and W. Geiselmann. Cryptanalysis of polly cracker. *IEEE Transactions on Information Security*, 48(11):2990–2991, 2002.

[49] R. Steinwandt, W. Geiselmann, and R. Endsuleit. Attacking a polynomial-based cryptosystem: Polly-cracker. *Int. Jour. Information Security*, 1:143–148, 2002.

[50] R. Steinwandt and M.I.G. Vasco. Chosen ciphertext attacks as common vulnerability of some group- and polynomial-based encryption schemes. *Tatra Mountains Mathematical Publications*, 33.

[51] N. Taslaman. Private key extension of Polly Cracker cryptosystems. *Comp. Sci. J. of Moldova*, 16:117–132, 2008.

[52] Y. Tsuchimoto. Preliminaries on Dixmier conjecture. *Mem. Fac. Sci. Kochi Univ. (Math.)*, 24:43–59, 2003.

[53] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, Cambridge, UK, 2nd edition, 2003.