Computing Splitting Fields and Galois Groups over Local Function Fields

Dissertation

Von der Fakultät für
Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

zur Erlangung des akademischen Grades
Doktor der Naturwissenchaften
Dr. rer. nat

vorgelegt von

Anthoula Zervou

Betreuer: Prof. Dr. Jürgen Klüners

Paderborn, 2023

Computing Splitting Fields and Galois Groups over Local Function Fields

Dissertation

Submitted to
the Faculty of
Computer Science, Electrical Engineering and Mathematics at
Paderborn University

in Fulfillment
of the Requirements for the Degree
Doctor of Natural Sciences
Dr. rer. nat.

by

Anthoula Zervou

Supervisor: Prof. Dr. Jürgen Klüners

Paderborn, 2023

This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project number KL 1424/10-1.

Dissertation by Anthoula Zervou (Anthi Zervou)
Supervisor: Prof. Dr. Jürgen Klüners

Doctoral Commission:
Prof. Dr. Jürgen Klüners (Supervisor and Referee)
Prof. Dr. Claus Fieker (Referee)
Prof. Dr. Igor Burban
Prof. Dr. Martin Kolb
Jun. Prof. Thomas Berger

Submission Date: September 29, 2023
Defense Date: December 15, 2023

◇ Ἓν οἶδα ὅτι οὐδὲν οἶδα (Σωκράτης)

[I know nothing except the fact of my ignorance. – Socrates]

◇ Μέτρον ἄριστον (Κλεόβουλος)

[Everything in moderation. – Cleobulus]

◇ Έχεις τα πινέλα, έχεις τα χρώματα, ζωγράφισε τον παράδεισο και μπες μέσα. (Νίκος Καζατζάκης)

[You have your brush, you have your colors, you paint the paradise, then in you go.
– Nikos Kazantzakis]

◇ Σα δεν φτάσει ο άνθρωπος στην άκρη του γκρεμού, δεν βγάζει στην πλάτη του φτερούγες να πετάξει. (Νίκος Καζατζάκης)

[If the man does not reach the edge of the cliff, he cannot grow wings to fly.
– Nikos Kazantzakis]

◇ Your destiny is determined by the choices you make. Choose now and choose well.
– Tony Robbins

◇ A man's usefulness depends upon his living up to his ideals in so far as he can.
– The Manhood stone tablet. (One of four tablets which stand on both sides of the Theodore Roosevelt Sculpture on Theodore Roosevelt Island.)

◇ It is hard to fail, but it is worse never to have tried to succeed.
– The Manhood stone tablet. (One of four tablets which stand on both sides of the Theodore Roosevelt Sculpture on Theodore Roosevelt Island.)

Acknowledgments

I would like to express my sincere gratitude to the individuals who played a pivotal role in the completion of this dissertation. Their unwavering support, guidance, and encouragement have been invaluable throughout this journey.

First and foremost, I express my deepest gratitude to my supervisor, Prof. Dr. Jürgen Klüners. Your support, guidance, wisdom and expertise have been instrumental in the completion of this research. Jürgen, your dedication to academic excellence and your insightful feedback have shaped this dissertation and my academic journey in profound ways. Thank you for giving me this opportunity.

Next, I would like to thank the members of the doctoral commission, Prof. Dr. Claus Fieker, Prof. Dr. Igor Burban, Prof. Dr. Martin Kolb, and Jun. Prof. Thomas Berger who have taken the time to review and evaluate this research. In particular, I would like to extend my appreciation to the external referee Prof. Dr. Claus Fieker for his constructive feedback and the thorough examination of my dissertation.

Furthermore, I sincerely want to thank Markus Kirschmer who generously shared his time, experience, and insights in answering my Magma-related questions. Markus, thank you for your positive spirit and the funny times we shared together. I want also to express my heartfelt gratitude to my office mate, Raphael, for his overall support and funny moments during the Ph.D. journey.

Moreover, I would like to thank my friends for their encouragement and understanding when I did not have enough time. Especially, I want to express my deepest gratitude to Mr. Manolis for helping me get myself back on track and guiding me in my pursuit of self-discovery.

Additionally, I want to acknowledge the members of my family who supported and constantly encouraged me throughout this academic journey. Particularly, my deepest thanks go to my twin sister, Marianna, for her emotional encouragement and understanding throughout the process. Your belief in my abilities and your unwavering support were essential in helping me overcome the challenges that came my way.

Last but not least, I would like to express my deepest and heartfelt gratitude to the person who has supported me the most during this academic journey. Manoli, your support, love, and encouragement sustained me during the challenging phases of this academic expedition. I deeply appreciate that we shared the highs and lows of this academic pursuit together. Your positive spirit and understanding have been my anchor in times of uncertainty. Thank you for everything!

I want to conclude these acknowledgments with my mother, Arsinoi, whose love, resilience, and the way she raised me have played a significant role in shaping the person I am today. I would like to remember and honor the memory of my beloved mother, who, though no longer with us, continues to inspire and motivate me every day. Her unwavering support and belief in my abilities played a pivotal role in my pursuit of knowledge and this academic achievement. I dedicate this work to her memory, with gratitude for her boundless encouragement and the sacrifices she made to pave the way for my education.

Anthi Zervou
Eindhoven, October 2023

Abstract

In this thesis, we develop an algorithm for computing Galois groups over local function fields. We use a variation of a splitting field approach by taking advantage of the fact that all local function fields are isomorphic to a Laurent series ring over a finite field. This idea reduces the complexity of the necessary factorizations of polynomials. By combining this with ramification polygons used in Greve's PhD-thesis for computing Galois groups over p-adic fields, we got further improvements.

All algorithms presented in this thesis have been implemented in the computer algebra system Magma. Moreover, the algorithm for computing the minimal polynomial of elements in local function fields has been improved. This speeds up the factoring algorithm of polynomials over local function fields.

As a result, we get the first implemented algorithm for computing Galois groups over local function fields. We demonstrate the efficiency by providing a large number of examples, where the resulting splitting fields have degrees up to 522240.

Zusammenfassung

In dieser Doktorarbeit wird ein Algorithmus zum Berechnen von Galoisgruppen über lokalen Funktionenkörpern entwickelt. Dabei wird eine Variante des Zerfällungskörper-Algorithmus verwendet und ausgenutzt, dass lokale Funktionenkörper isomorph zu einem Laurentreihenring über endlichen Körpern sind. Diese Idee reduziert die Komplexität der benötigten Polynomfaktorisierungen. Die Kombination dieses Ansatzes mit Verzweigungspolygonen, welche in der Doktorarbeit von Greve zur Berechnung von Galoisgruppen über p-adischen Körpern eingeführt wurden, liefert eine weitere Verbesserung.

Alle in der Arbeit vorgestellten Algorithmen wurden im Computeralgebrasystem Magma implementiert. Darüber hinaus wird der Algorithmus zum Berechnen von Minimalpolynomen in lokalen Funktionenkörpern verbessert. Dies beschleunigt die Berechnung von Zerfällungskörpern über diesen Körpern.

Insgesamt ergibt sich die erste Implementierung eines Algorithmus zum Berechnen von Galoisgruppen über lokalen Funktionenkörpern. Anhand von einer Vielzahl von Beispielen wird die Effizienz des Ansatzes demonstriert. So wird z.B. ein Zerfällungskörper vom Grad 522240 bestimmt.

Contents

Introduction

Galois theory is a connection between field and group theory. It gives a bijective correspondence between intermediate fields of a Galois extension and the subgroups of its Galois group. One of the driving forces of mathematics for hundreds of years was the solvability of polynomial equations. A historical result of Galois theory asserts that for every $n \geq 5$ there are polynomials of degree n which are not solvable by radicals. On the contrary, all roots of a polynomial up to degree four can be written by radicals. Galois theory attaches a permutation group to each polynomial. The solvability of a polynomial by radicals is then equivalent to the solvability of that group. Since all permutation groups up to degree four are solvable, we get the aforementioned result. It is well known that symmetric and alternating groups of degree at least five are not solvable and in every degree we can find polynomials over the rationals with these Galois groups.

Galois theory is divided into the direct and the inverse problem. In the direct problem we start with a given polynomial and we ask for its Galois group. The inverse problem concerns whether or not a given finite group is realizable as Galois group of some finite extension over a specific base field. Both direct and inverse problems depend on the given base field.

In this thesis we are interested in the direct problem. In particular, we implement an algorithm for the computation of the Galois group of a given polynomial over local function fields of prime characteristic, i.e the Laurent series field $\mathbb{F}_q((t))$ over the finite field with q elements. The above algorithm uses the splitting field approach, and so we implement an algorithm for the computation of the splitting field of a given polynomial as well. For the implementation of the latter, we use a variation of the basic approach. The new ingredient in the local function field case is that all local function fields are isomorphic to Laurent series rings over finite fields. For this reason, a splitting field approach, which is usually not very practical, looks very promising in the local function field case. Moreover, we implement an algorithm for the minimal polynomial computation in order to generate an extension which improves the behavior of the factorization code, which plays a vital role in our splitting field algorithm.

Known results

Classically, the first algorithms for computing Galois groups have been developed for rational polynomials. The most efficient algorithms for practical examples are based on an algorithm introduced by Stauduhar [48]. These algorithms use approximations of the roots in some field containing the rational numbers, e.g the complex numbers or suitable p-adic extensions [10, 13]. Certainly, on a computer we can only use approximations of complex or p-adic numbers. The implementations of the p-adic versions used some (proven) error estimates and therefore the computed results have been mathematically proven. These methods have been extended to other fields like number fields, global function fields and function fields [27] over the rationals or number fields, respectively. All these algorithms are using the fact that we can compute approximations of the roots. In case that the base field K is a p-adic field or a local function field, the algorithm based on ideas of Stauduhar is not applicable since we do not have access to the roots. It is not trivial to write down a field (like the complex numbers in the rational case) which contains all the roots of the given polynomial.

It is known that the Galois group of a polynomial of degree n is a subgroup of the symmetric group S_n. We can speak about Galois groups of both irreducible or reducible polynomials. For an irreducible polynomial, its Galois group is a transitive subgroup of S_n. We can compute the

Galois group of a polynomial by computing the splitting field of the polynomial. Over the splitting field the given polynomial splits into linear factors. This enables us to compute the Galois group represented as a permutation group on the roots of the polynomial. This approach is called the basic approach, and we emphasize that it is efficient when the splitting field is of a small degree. This is clear from a theoretical point of view, and can also be demonstrated with explicit examples. Nevertheless, the basic approach has the advantage that it can be applied to many base field cases.

Until a few years ago, only the basic algorithm was known to attack this problem over local function fields. In 2010 Christian Greve in his PhD-thesis [15] developed non-trivial algorithms for computing the Galois group of Eisenstein polynomials, by using the ramification polygon introduced by Romano [41]. For Galois extensions the ramification polygon gives enough information to compute all the higher ramification groups, whereas in the non-Galois case we only get partial information. Greve gave different algorithms if the ramification polygon consists of one or two segments. These algorithms are very satisfying in the one segment case. In the two segment case he got a description of the Galois group by using the canonical class. For the case of p-adic fields he implemented the Galois group computation in both cases, but it was never published outside of his thesis. In 2017 Jonathan Milstead in his PhD-thesis [29] uses the results of Greve as a first step and then proceeds with the absolute resolvent method. His algorithm to compute the Galois group of an Eisenstein polynomial over p-adic fields has been successfully applied to polynomials whose ramification polygons have two or more segments.

It is known that p-adic fields have been studied extensively and there exist implementations in several computer programs such as Magma [1]. On the other hand, this is not true for the case of local function fields. The implementations are limited and much weaker than the ones of the p-adic case. The aim of this thesis is to develop and implement algorithms for the computation of splitting fields and Galois groups of polynomials over local function fields i.e $\mathbb{F}_q((t))$.

To begin with, we attack this problem by using the basic approach, which is based on the successive extensions of local function fields. This means that we obtain the splitting field by adjoining some roots of the given polynomial to the base field. In particular, let $K := \mathbb{F}_q((t))$, then we construct a tower of fields

$$K := K_0 \leqslant K_1 \leqslant \ldots \leqslant K_{\ell-1} \leqslant K_\ell := \mathrm{Spl}(f)$$

such that $K_i = K_{i-1}(\alpha_i)$, where α_i is a new root of $f \in K[X]$. We successively adjoin a root of an irreducible factor of f to the current extension field, and then we refactor the polynomial over the new extension field. We repeat this process until the polynomial $f \in K[X]$ completely factors.

However, the representation of the splitting field by successive extensions is not practical in all cases and heavily depending on the degree of the splitting field. Since f has at most $\deg(f)$ roots, the construction will require at most $\deg(f)$ extensions. The bigger the degree $[K_i : K]$ is, the more expensive becomes the factorization of the corresponding polynomials. This is true even if we take into account that we factor polynomials of smaller degree by using the already obtained factorizations of the given polynomial f over K_{i-1}. Moreover, when we implemented the algorithm based on the basic approach in Magma [1], it turned out that Magma has a lot of problems with polynomials given over extensions of K. Let us describe this situation. As before $K = \mathbb{F}_q((t))$ and $f \in K[X]$ is an irreducible polynomial. Then $K_1 := K[X]/\langle f\rangle$ and the field K_1 is represented as the residue class ring of the polynomial ring $K[X]$. Experiments showed that the running time of the factoring algorithm in Magma over K_1 was much slower than expected and furthermore there were many examples where the code was crashing. We emphasize that the situation is much nicer when considering polynomials over K.

Splitting Field: A Variation of the Basic Approach

The arithmetic over $K := \mathbb{F}_q((t))$ is much less computationally expensive than over $\mathbb{F}_q((t))[X]/\langle f\rangle$, let alone over $K_i = K_{i-1}[X]/\langle f_i\rangle$, where f_i is an irreducible factor of f such that $f_i(\alpha_i) = 0$. In order to avoid those computational problems we replace the structure of the complicated successive extension field of the form $K_i = K_{i-1}[X]/\langle f_i\rangle$ with a local function field of the form $\mathbb{F}_{q_i}((u_i))$, where $\mathbb{F}_{q_i}$ is the constant field extension of K_i and u_i is a uniformizing element of K_i, in every step. This way, we can apply the algorithms for this situation, which are more efficient (and more stable in Magma) compared to the implementations over K_i.

Using the above idea, it is important to note that we have to determine how the element t is embedded into the field $\mathbb{F}_{q_i}((u_i))$. That is to say, we should express t as an element of $\mathbb{F}_{q_i}((u_i))$. Furthermore, due to the fact that the factorization algorithm over local fields outputs approximations modulo a power of the maximal ideal, it can lead to false results. Consequently, we need to choose the precision in $\mathbb{F}_{q_i}((u_i))$ and $\mathbb{F}_q((t))$ carefully.

Motivated by the above idea, we develop a variation of the basic approach for the splitting field computation over local function fields. It is clear that every K_i is a local function field with finite residue field, and thus, every field K_i is isomorphic to a field of formal Laurent series $L_i := \mathbb{F}_{q_i}((u_i))$. Therefore, we construct a tower of fields

$$K := \mathbb{F}_q((t)) = L_0 < L_1 < \dots < L_{\ell-1} < L_\ell := N = \mathrm{Spl}(f), \text{ where } L_i := \mathbb{F}_{q_i}((u_i)).$$

The main advantage in this way is that in every intermediate step we work with the field $\mathbb{F}_{\tilde{q}}((u))$ instead of working with an extension of $\mathbb{F}_q((t))$ by adjoining a root α of the initial polynomial. On the other hand, we should determine how u_{i-1} is embedded into L_i for every $1 \le i \le \ell - 1$ where $u_0 = t$ and the needed precision in L_i for every $1 \le i \le \ell$.

Denote by $e_i = e(K_i/K_{i-1})$ the ramification index of K_i/K_{i-1} and by π_{K_i} a uniformizing element of $\mathcal{O}_{K_i}$. We inductively get

$$K_i = K_{i-1}[X]/\langle f_i \rangle = \mathbb{F}_{q_{i-1}}((u_{i-1}))(\alpha_i) \cong \mathbb{F}_{q_i}((u_{i-1}))[X]/\langle g_i \rangle = \mathbb{F}_{q_i}((u_{i-1}))(\pi_{K_i}) \cong \mathbb{F}_{q_i}((u_i)),$$

where $q_i = q_{i-1}^{\mathfrak{f}_i}$ with $\mathfrak{f}_i = \mathfrak{f}(K_i/K_{i-1})$ and g_i is the minimal polynomial of π_{K_i} over $\mathbb{F}_{q_i}((u_{i-1}))$ of degree e_i. Then for every $1 \le i \le \ell$ we define the isomorphism

$$\Phi_i : \mathbb{F}_{q_i}((u_{i-1}))(\pi_{K_i}) \to \mathbb{F}_{q_i}((u_i)),\ \Phi_i|_{\mathbb{F}_{q_i}} = \mathrm{id},\ \pi_{K_i} \mapsto u_i,\ u_{i-1} \mapsto \Phi_i(u_{i-1}).$$

The determination of $\Phi_i(u_{i-1})$ can be done by linear algebra and is explained in Proposition 3.2.5.

By extending Φ_i to the polynomial ring via $X \mapsto X$ we can represent all of our factors in the new presentation, and the next factorization steps can be done over this field. Since the right hand field is again a Laurent series field over a finite field, we can use the factoring algorithm over this field, which looks like the base field, and therefore the coefficient field of the polynomial ring does not become more complicated during this inductive procedure. Nevertheless, it is clear that some needed u_{i-1}-precision O_{i-1} will be replaced by something like a u_i-precision of order $e_i \cdot O_{i-1}$. Overall, the steps of our approach are summarized in Remark 3.2.8.

Determining the precision by the above strategy can lead to very large values of it, which negatively affects the time complexity performance of the algorithm. After studying this problem, we develop a new approach for the choice of precision in every step by using information obtained by the ramification polygon of the polynomial. Experiments yielded better values for the precision needed in every step, (see Heuristic Assumption 3.2.10).

Having computed the splitting field N in this presentation, we also have access to all the roots of our given polynomial f, which means that an approximation of a root of f is of the form $\sum_{i=e(N/K)}^{\tilde{B}} a_i u_\ell^i$, with $a_i \in \mathbb{F}_{q_\ell}$, where $\tilde{B}$ denotes the resulting u_ℓ-precision of N.

Factorization

It is clear that the factorization of a polynomial plays a vital part in the splitting field computation using the naive approach. While making experiments on the code and analyzing its time complexity, we noticed that in many cases the factorization took a lot of time. In fact, this behavior was even worse while performing computations over local function fields with large precision.

The factorization algorithm is available for polynomials over series rings defined over finite fields in Magma [1]. When we factorize a polynomial over $\mathbb{F}_q((t))$ we can use the optional parameter “Extensions”. By setting it to true, an extension for each irreducible factor of the polynomial is returned together with certificates like its uniformizing element and its minimal polynomial.

This parameter is a critical part in our algorithm, since at several places in our algorithm we need this minimal polynomial in order to generate an extension with a nicer polynomial than the given one. Experiments showed that the running time for this minimal polynomial computation was very slow compared to that of other parts. Thus, it was necessary to improve this part.

In case of fields of small positive characteristic, Magma computes the minimal polynomial of an element β by solving a linear system of equations. This is slow for many examples, especially when we need large precision. In order to improve this part, we use a better method for computing a generating polynomial of a desired extension. This is based on the representation of a polynomial by using higher traces (power sums) of the roots and vice versa. Using the first d (or $2d$) higher traces of the roots of a degree d-polynomial, we reconstruct the polynomial from them. We note that the Newton Relations provide a straightforward way to make these conversions in large characteristic or characteristic 0. In [3, 2] the authors presented much better approaches for the conversions between a monic polynomial and the power sums of its roots, which we discuss in Chapter 4. In the small positive characteristic case, given $f \in K[X]$ of degree d such that all of its roots have multiplicities less than p, the polynomial f can be constructed by the first $2d$ power sums of its roots, (see Proposition 4.2.3, 2). This can be done by using Algorithm 11. An important role in this algorithm plays the computation of the minimal polynomial of the linear recurrent sequence of the power sums of the roots of f. For its computation, we use the Berlekamp-Massey algorithm (see Algorithm 14). Because of divisions in Euclidean-type algorithms we face a lot of precision losses during our process. The critical point here is the determination of the precision at the beginning of the computation of the minimal polynomial of a linear recurrent sequence (Section 4.4).

A further improvement is the direct computation of the minimal polynomial of $\tilde{\beta} = \frac{1}{t^{\tilde{d}}}\beta$ instead of computing it by using the minimal polynomial of β. This is done by taking into account its denominators when computing the higher traces, see Section 4.5.2.

In Table A.16, we give a list of examples in which we compare the time required to factorize them using the "Extensions"-parameter by employing Magma's algorithm and our algorithm, respectively. We remark that in the cases of large precision our algorithm is faster as indicated in Table A.16. In particular, that table includes a polynomial of degree 2 over $\mathbb{F}_{2^2}((t))$ with t-precision 26880 in which the running time for its factorization reduced from 41.22 to 0.29 seconds.

Splitting Field: Combined Approach

To sum up, we have implemented an algorithm for computing the splitting field of a polynomial over a local function field which is based on the idea that every local function field with finite residue field $\mathbb{F}_{\tilde{q}}$ is isomorphic to the field of formal Laurent series $\mathbb{F}_{\tilde{q}}((u))$. Our next goal is to determine the splitting field and the Galois group of a polynomial over local function fields by using the results of Greve and Pauli in [16] and of Greve's doctoral research in [15].

They thoroughly study the case of tamely ramified extensions. Furthermore, in the case of Eisenstein polynomials, by using the ramification polygon, they compute a canonical tower of subfields of the given extension, which reduces the Galois group computation to a p-problem ([16], Theorem 9.1 or Theorem 2.4.1). Additionally, Theorem 2.4.1 can be used in order to improve the splitting field algorithm. To further explain, the above theorem explicitly provides the existence of a subfield T of the splitting field N of the given polynomial g such that N/T is a p-extension. Thus, it can be divided in a tower of elementary abelian relative extensions. On the whole, for many examples this theorem gives us an important part of the splitting field with a little effort.

Combining this result with our approach to compute the splitting field, we obtain a further improved splitting field algorithm, which we analyze in Section 5.1. The precision in this approach is determined by Heuristic Assumption 5.1.2. Moreover, we investigate the case of non-Eisenstein irreducible polynomials and develop an algorithm for attacking those cases. For a fuller treatment of this case, we divide it into three sub-cases, which we thoroughly examine so as to implement the corresponding algorithms. Especially, the three sub-cases are: the totally ramified case (i.e. $e = \deg(f)$ & $\mathfrak{f} = 1$), the unramified case (i.e. $\mathfrak{f} = \deg(f)$ & $e = 1$) and the mixed case (i.e. $e > 1$ & $\mathfrak{f} > 1$), where e stands for the ramification index and $\mathfrak{f}$ denotes the inertia degree.

Having Greve's results in [15] as an incentive, we deal with some special cases separately. One of them is the case of tamely ramified extensions over local function fields, in which we create an algorithm based on the results analyzed in [15] (Section 1.9). Another special case is that of Eisenstein polynomials whose ramification polygon consists of one segment, which has been extensively studied over p-adic fields by Greve and Pauli in [16]. Their results (Section 2.3.1) can be applied for general local fields, and so for local function fields. Considering them over local function fields, we implement an algorithm for this case as well. Although the results of Greve and Pauli can be applied to general local fields, their implementation exists only over p-adic fields.

In Section A.1, we give some examples in which we compare the running times between the aforementioned approach and the so-called “A Variation of the Basic Approach”. We remark that those examples show that in many cases the computation of the field T yields a big step towards the direction of the splitting field, and thus results in speeding up the whole process. We note that in the last example of characteristic 3 in Table A.2 the splitting field is of degree 30618 and its running time for its computation reduced from 22755.7 to 6711.0 seconds.

Splitting Field: Improved Approach

One drawback of our method by using that $K_i \cong \mathbb{F}_{q_i}((u_i))$ is that the needed precisions in extension fields are increasing. When we start with an Eisenstein polynomial $f \in K[X]$, the irreducible factors of f over extension fields K_i are far away of being Eisenstein. This leads to the problem that we need more precision in order to get proven results. The necessary precision can be reduced, when we replace the irreducible factors of f over K_i by Eisenstein polynomials. In particular, in every intermediate step we construct the extension by using the corresponding Eisenstein polynomial g_i of the current candidate factor f_i such that

$$K_{i-1}[X]/\langle f_i \rangle \cong K_{i-1}[X]/\langle g_i \rangle.$$

As we have already described, those Eisenstein polynomials are the minimal polynomials of the computed uniformizing elements, and those can be computed by using the optional parameter “Extensions” in the factorization algorithm provided by Magma. Certainly, the irreducible factor and the corresponding Eisenstein polynomial generate the same field extension, since $\mathrm{Spl}(f_i/K_{i-1}) = \mathrm{Spl}(g_i/K_{i-1})$. Therefore, we can use the much nicer Eisenstein polynomial for the rest of the splitting field computation. The mathematical realization of this idea is easy, but the implementation needs a lot of attention, which is described in Section 5.4.

In Section A.2, we give some tables of examples in which we compare the running times between the aforementioned approach and the so-called “Combined Approach”. We remark that in the last example of characteristic 3 in Table A.4 the splitting field is of degree 104976 and its running time for its computation reduced from 3663.1 to 669.8 seconds.

Galois Group Algorithm

Having computed the splitting field, we can proceed to its Galois group. Let $f \in K[X]$, where $K := \mathbb{F}_q((t))$, be a polynomial of degree n with roots $\alpha_1, \dots, \alpha_n$. The Galois group of f over K is defined to be the Galois group of $\mathrm{Spl}(f)$ over K. We can view any automorphism on the splitting field as a permutation of the roots of the polynomial. In our case, $\mathrm{Spl}(f) = \mathbb{F}_{q_\ell}((u_\ell))$ and so in order to represent an element σ of $\mathrm{Gal}(N/K)$, we need to know the image $\sigma(u_\ell)$ and how it acts on $\mathbb{F}_{q_\ell}/\mathbb{F}_q$. Especially, the computation of $\sigma(u_\ell)$ is not trivial. In case of a constant field extension, σ is the Frobenius automorphism (see Remarks 6.1.3 and 6.1.4). For the general case, i.e. non constant field extensions, it holds that $\sigma(w) = w$, where $\mathbb{F}_{q_\ell}^\times = \langle w \rangle$ and we determine $\sigma(u_\ell)$ by solving the equation $\sigma(\alpha_i) = \alpha_j$, for some $1 \leq i \neq j \leq n$. By using linear algebra, that equation can be solved by writing down and solving a suitable system of equations. This can be found in Proposition 6.1.8 for the case $p \nmid v_u(\alpha_i)$. Moreover, we can construct those equations by using the partitions of suitable numbers which is presented in Proposition 6.1.15.

For the wildly ramified case i.e. $p \mid v_u(\alpha_i)$ similar arguments can be applied and the computation of $\sigma(u_\ell)$ is given in Proposition 6.1.34. More details and the proofs of them can be found in Section 6.1.

Considering all the above, we combine the results in order to create an algorithm for the Galois group over local function fields computation. Especially, for the aforementioned special cases we employ the corresponding algorithms for the desired Galois group computation. Section A.3 show that it is possible to compute the Galois group of any separable polynomial f given over some local function field K.

Outline

In conclusion, in this thesis we develop and implement algorithms for the computation of splitting fields and Galois groups of polynomials over local function fields. All algorithms are implemented in the computer algebra system Magma [1]. We emphasize that our algorithms have no restrictions on the given polynomial. They can be applied to Eisenstein polynomials, but also to non-Eisenstein irreducible (see Tables A.11, A.12) or reducible polynomials (see Tables A.13, A.14). In particular, in the non-Eisenstein irreducible case, we have computed a Galois group of order 24192 in characteristic 2 as indicated in Table A.11. Moreover, we note that our algorithms have been successfully applied to many Eisenstein polynomials whose ramification polygon consists of two or more segments. Particularly, a number of those examples correspond to the three segments case as shown in Tables A.5 and A.6. Also, our method has been successfully applied to polynomials whose ramification polygon consists of four segments. Especially, in Table A.5 there are two polynomials of four segments with Galois groups of order 2048. In Appendix A we give tables of examples in characteristic 2 and 3, respectively. Especially, the tables in Section A.3 show that it is possible to compute Galois groups of higher orders. In particular, the examples show that many of them have Galois groups of order more than one thousand. Remarkably, we have computed a Galois group of order 12960 in characteristic 3 (see the last example of Table A.6) for an Eisenstein polynomial with 2 segments in its ramification polygon. Moreover, for the three segments case, we have computed a Galois group of order 6144 in characteristic 2 as indicated in Table A.5. Furthermore, we have computed a Galois group of order 4960 in characteristic 2 (see the last example of Table A.7), and of order 2106 in characteristic 3 (see the last example of Table A.8) for polynomials whose ramification polygon consists of one segment. Additionally, for the tamely ramified case, we have computed a Galois group of order 26364 in characteristic 2 (see last example in Table A.9). Note that in Table A.15 we provide a list of strongly Eisenstein polynomials, where the resulting splitting fields have degree up to 522240. We remark that our implementations for computing the splitting field as well as the Galois group of a given polynomial over local function fields are the first of their kind.

This dissertation is devoted to the study of splitting fields and Galois groups of polynomials over local function fields. The dissertation consists of six chapters and an appendix.

◇ **Chapter 1 (Preliminaries)** describes the basic facts (and theory) about local function fields. We also study the splitting fields and Galois groups of tamely ramified extensions.

◇ **Chapter 2 (Ramification Polygon & Applications)** discusses two invariants of the extensions generated by an Eisenstein polynomial: the ramification polygon and the residual (associated) polynomials. We also give the relevant material for splitting fields and Galois groups of Eisenstein polynomials whose ramification polygon consists of one segment. This case has been analyzed by Greve in Chapter 5 in his PhD-thesis [15] and by Greve and Pauli in [16]. We conclude this chapter by giving the p-Reduction Theorem (Theorem 2.4.1) which will be used for dealing with the general case of Eisenstein polynomials whose ramification polygon has more than one segments.

◇ **Chapter 3 (Splitting Field over Local Function Fields)** describes our new algorithm for the splitting field computation over local function fields. In the beginning we briefly describe the basic approach of splitting field computation. Then the rest of the chapter delves into the description of our approach “A Variation of Basic Approach” (Section 3.2) which is based on the idea that every local function field with finite residue field $\mathbb{F}_{\tilde{q}}$ is isomorphic to the field of formal Laurent series $\mathbb{F}_{\tilde{q}}((u))$.

◇ **Chapter 4 (Factorization)** discusses a method for the computation of a generating polynomial of an extension over local function fields. In particular, we use well known tricks in small characteristic described in [3, 2]. The situation here is not completely analogous to [3, 2], since we only have approximations in the base field. For fields of small positive characteristic we can construct a polynomial $f \in K[X]$ of degree d by using the first $2d$ power sums of its roots under some extra assumption for its roots, (see Proposition 4.2.3, 2), which is studied in Section 4.3. In Section 4.4, we study the computation of a minimal polynomial of a linear recurrent sequence which is based on the Berlekamp-Massey algorithm, (see Algorithm 14). In Section 4.5, we describe our algorithm for the computation of a generating polynomial of an extension.

We note that the aforementioned algorithm can be integrated into the original factorization algorithm provided by Magma [1].

- ◇ **Chapter 5 (Splitting Field: Combined Approach)** is devoted to develop an improved approach of computing the splitting field of a given polynomial. To be more precise, in this approach we combine our approach developed in the section "A Variation of the Basic Approach" (Section 3.2) with the one developed by Greve in his thesis [15], which we study in Sections 1.9, 2.3.1, 2.4. We call it "Combined Approach". The next improvement of splitting field computation is presented in Section 5.4. This approach is based on the idea that in every intermediate step we construct the extension by using the corresponding Eisenstein polynomial of the current candidate factor of the initial polynomial. Certainly, the irreducible factor and the corresponding Eisenstein polynomial generate the same field extension. We refer to this approach as "Improved Approach".

- ◇ **Chapter 6 (Computation of Galois Group over Local Function Fields)** provides a detailed exposition of the theory for our method that computes the Galois group of a polynomial over a local function field, which is studied in Section 6.1. In addition, the method is encapsulated in an algorithm, which is Algorithm 31. Overall, it summarizes the results into an algorithm. In particular, for the special cases, it employs the algorithms for tamely ramified and one segment cases, respectively.

- ◇ **Appendix A (Examples)** contains tables of examples. In Section A.1 we give tables of examples in which we compare the two approaches -"A Variation of the Basic Approach", (Ver1), and "Combined Approach", (Ver2),- for the splitting field computation. In Section A.2 we give examples in which we compare the two approaches -"Combined Approach", (Ver2), and "Improved Approach", (Ver3),- for the splitting field computation. In Section A.3 we provide tables of examples in which we have successfully computed the splitting fields and Galois groups. In Section A.4 we give a table of examples in which we compare the running time needed for their factorization using "Extensions"-parameter by employing Magma's algorithm and our algorithm, respectively.

Chapter 1

Preliminaries

In this chapter we will mention some basic facts about local field theory and local class field theory. For further details on the topic we refer the reader to the relevant sections in [9, 19, 35].

1.1 Local Fields

We give a short survey of the theory of local fields. In order to define local fields we provide some notions of valuation theory.

Definition 1.1.1
Let K be a field. A map $|\cdot| : K \mapsto \mathbb{R}$ is called a ***absolute value*** *if for every $x, y \in K$ we have*

(i) $|x| \geq 0$, *and* $|x| = 0 \Leftrightarrow x = 0$,

(ii) $|xy| = |x||y|$,

(iii) $|x + y| \leq |x| + |y|$.

Definition 1.1.2
A valuation $|\cdot| : K \mapsto \mathbb{R}$ is called ***non-Archimedean (or ultrametric)*** *if and only if it satisfies the* ***strong triangle inequality***

$$|x + y| \leq \max\{|x|, |y|\} \tag{1.1}$$

for every $x, y \in K$, instead of the condition (iii) of Definition 1.1.1. Otherwise, it is called ***Archimedean****.*

Example 1.1.3
The usual absolute value on the real numbers $\mathbb{R}$ is an Archimedean absolute value.

Remark 1.1.4
The strong triangle inequality immediately implies that if $|x| \neq |y|$, then we have the equality $|x + y| = \max\{|x|, |y|\}$.

Definition 1.1.5
A map $\upsilon : K \to \mathbb{Q} \cup \{\infty\}$, where K is a field, is called ***(exponential) valuation*** *of K if for every $a, b \in K$ we have*
1) $\upsilon(\alpha) = \infty \Leftrightarrow \alpha = 0$,
2) $\upsilon(\alpha\beta) = \upsilon(\alpha) + \upsilon(\beta)$,
3) $\upsilon(\alpha + \beta) \geq \min\{\upsilon(\alpha), \upsilon(\beta)\}$.

If $\upsilon(K^\times) = \{0\}$, where $K^\times = K \setminus \{0\}$, then υ is called trivial exponential valuation. In addition, similarly to Remark 1.1.4, it is easy to show that if $\upsilon(\alpha) \neq \upsilon(\beta)$, then $\upsilon(\alpha + \beta) = \min\{\upsilon(\alpha), \upsilon(\beta)\}$.

Example 1.1.6
Let $\mathbb{Q}_p$ be the p-adic numbers, where p is a prime number. Let $\alpha \in \mathbb{Q}_p$, then

$$\alpha = \sum_{i=-m}^{\infty} a_i p^i = a_{-m}p^{-m} + \cdots + a_{-1}p^{-1} + a_0 + a_1 p + a_2 p^2 + \cdots, \quad \text{where } m \in \mathbb{Z} \text{ and } 0 \leq a_i < p.$$

The p-adic norm of a rational number α is given by

$$| \ |_p : \mathbb{Q} \to \mathbb{R}, \ \ \alpha \mapsto |\alpha|_p := p^{-\upsilon_p(\alpha)},$$

where $\upsilon_p(\alpha)$ is the maximal integer such that $p^{\upsilon_p(\alpha)}$ divides α. In particular, υ_p is a map $\upsilon_p : \mathbb{Q} \to \mathbb{Z} \cup \{\infty\}$, which satisfies the properties:
1) $\upsilon_p(\alpha) = \infty \Leftrightarrow \alpha = 0$,
2) $\upsilon_p(\alpha\beta) = \upsilon_p(\alpha) + \upsilon_p(\beta)$,
3) $\upsilon_p(\alpha + \beta) \geq \min\{\upsilon_p(\alpha), \upsilon_p(\beta)\}$,
where $x + \infty = \infty$, $\infty + \infty = \infty$ and $\infty > x$, for every $x \in \mathbb{Z}$.
Then we can show that the p-adic norm is a non-Archimedean valuation.

Let K be a field. Two absolute values $|\cdot|_1$ and $|\cdot|_2$ on K are said to be *equivalent* if and only if there is a real number $s \in \mathbb{R}_{>0}$ such that $|x|_1 = |x|_2^s$ for every $x \in K$. Two exponential valuations υ_1, υ_2 of K are equivalent if $\upsilon_1 = s\upsilon_2$, with $s \in \mathbb{R}$ and $s > 0$.

For every exponential valuation υ we can determine an associated absolute values $|\cdot|$, by defining $|x| = c^{-\upsilon(x)}$, where c is some fixed real number. Also, given two equivalent exponential valuations, then the associated absolute values are equivalent. This means that if $\upsilon_1 = s\upsilon_2$, then $|\cdot|_1 = |\cdot|_2^s$, with $s \in \mathbb{R}$ and $s > 0$.

Definition 1.1.7
A field K with a valuation on it is called a valued field.

Definition 1.1.8
Let K be a field with exponential valuation υ_K and its associated non-Archimedean absolute value $|\cdot|$. We define the ring of integers of K,

$$\mathcal{O}_K = \{x \in K \,|\, \upsilon_K(x) \geq 0\} = \{x \in K \,:\, |x| \leq 1\},$$

the group of units,

$$\mathcal{U}_K = \{x \in K \,|\, \upsilon_K(x) = 0\} = \{x \in K \,:\, |x| = 1\},$$

and the unique maximal ideal of $\mathcal{O}_K$,

$$\mathcal{M}_K = \{x \in K \,|\, \upsilon_K(x) > 0\} = \{x \in K \,:\, |x| < 1\}.$$

The field K is the field of fractions of the ring of integers $\mathcal{O}_K$ of K. The quotient field $\underline{K} = \mathcal{O}_K/\mathcal{M}_K$ is called residue class field of K. For an element $\gamma \in K$, we denote by $\underline{\gamma}$ an element of $\underline{K}$, and is defined as the class $\underline{\gamma} = \gamma + \mathcal{M}_K \in \underline{K}$.

An exponential valuation is called ***discrete*** if $\upsilon : K \to \mathbb{Z} \cup \{\infty\}$. This means that the group $\upsilon(K^\times)$ is isomorphic to additive group $(\mathbb{Z}, +)$. In this case, we have that $\upsilon(K^\times) = s\mathbb{Z}$, with $s > 0$. It is called ***normalized*** if $s = 1$, that is $\upsilon(K^\times) = \mathbb{Z}$. In general, we can achieve a normalized valuation from the initial exponential one by dividing by s, and the normalized one has the same invariants $\mathcal{O}_K$, $\mathcal{U}_K$ and $\mathcal{M}_K$. Thus, without loss of generality we shall often assume that the exponential valuation is normalized.

Now we give the definition of a local field as Neukirch defined in Section 5, Chapter II in [35].

Definition 1.1.9
*A **local field** is a field which is complete with respect to a discrete, non-Archimedean absolute value and its residue class field is finite.*
For example, $\mathbb{Q}_p$ and $\mathbb{F}_q((t))$ are local fields.

For such local field K we denote the normalized exponential valuation by υ_K and the associated non-Archimedean absolute value by $|\cdot|_K$, which is defined by

$$|x|_K = q^{-\upsilon_K(x)},$$

where q is the cardinality of the residue class field $\underline{K}$ of K.

In the case of the global fields, we know that the global fields are precisely the finite extensions of either $\mathbb{Q}$ or $\mathbb{F}_p(t)$. It is worth pointing out that there is also a corresponding characterization for the case of local fields, which is given below.

Proposition 1.1.10 (Proposition (5.2), Chapter 2 in [35])
The local fields are precisely the finite extensions of the fields $\mathbb{Q}_p$ and $\mathbb{F}_p((t))$.

It is known that if υ_K is a discrete exponential valuation of K, then $\mathcal{O}_K$ is a principal ideal domain, that is an integral domain in which every ideal is principal. In addition to that, $\mathcal{O}_K$ has exactly one non-zero maximal ideal. Hence, $\mathcal{O}_K$ is a discrete valuation ring. See Proposition (3.9), [35] for more details. Therefore, the unique maximal ideal $\mathcal{M}_K$ of $\mathcal{O}_K$ is a principal ideal.

Definition 1.1.11
An element $\pi \in \mathcal{O}_K$ such that $\upsilon_K(\pi) = 1$ is called ***prime element or uniformizing element****. Every element $\pi_K \in K$ such that generates $\mathcal{M}_K$ is a prime element or uniformizing element of K. In particular, $\mathcal{M}_K = \pi_K \mathcal{O}_K$. It is worth noting that $\upsilon_K(\pi_K) = 1$.*

From now on, we write υ_K for the (exponential) valuation of K that is normalized such that $\upsilon_K(\pi_K) = 1$, where π_K is a uniformizing element in $\mathcal{O}_K$.

Proposition 1.1.12 (Proposition (5.3), [35])
The multiplicative group of a local field K admits the decomposition

$$K^\times = \langle \pi_K \rangle \times \mu_{q-1} \times U^{(1)},$$

where π_K is a prime element, $\langle \pi_K \rangle = \{\pi_K^\kappa | \, \kappa \in \mathbb{Z}\}$, $q = |\underline{K}|$, μ_{q-1} is the group of $q-1$-th roots of unity, and $U^{(1)} = 1 + \mathcal{M}_K$ is the group of principal units.

Example 1.1.13
Let $\mathbb{Q}_p$ be the p-adic number fields where p is a prime number, which is a local field. The local fields of characteristic 0 are precisely the finite extensions $K/\mathbb{Q}_p$ of $\mathbb{Q}_p$. The p-adic field is a valued field equipped with the p-adic norm $|\cdot|_p$, as we defined in the Example 1.1.6, and it is the completion of the rational field $\mathbb{Q}$ with respect to the p-adic norm. The p-adic integers, denoted by $\mathbb{Z}_p$ is the set of elements of $\mathbb{Q}_p$ such that they have non-negative p-adic exponential valuation υ_p, (see Example 1.1.6), that is

$$\mathbb{Z}_p = \{x \in \mathbb{Q}_p \,|\, \upsilon_p(x) \geq 0\} = \{x \in \mathbb{Q}_p \,:\, |x|_p \leq 1\}.$$

Thus, the ring of integers of $\mathbb{Q}_p$ is the ring of p-adic integers, that is $\mathcal{O}_{\mathbb{Q}_p} = \mathbb{Z}_p$.
The unique maximal ideal of $\mathbb{Z}_p$ is generated by p, this means that $\mathcal{M}_{\mathbb{Q}_p} = p\mathbb{Z}_p$.
Moreover, we have that the residue class field of $\mathbb{Q}_p$ is the finite field $\mathbb{F}_p$ consisting of p elements, that is $\underline{\mathbb{Q}_p} = \mathbb{Z}_p/p\mathbb{Z}_p \cong \mathbb{F}_p$.

The local fields of prime characteristic p are the Laurent series fields $\mathbb{F}_q((t))$, with $q = p^l$. We will focus on those fields in the next section.

1.2 Local Function Fields

Many authors have studied the case of local fields with finite residue class field of characteristic 0, for example see [14, 33]. In this section, we explicitly study the case of the local fields with finite residue field of prime characteristic, i.e $\mathbb{F}_q((t))$, with $q = p^l$, where p is a prime number.

Throughout the thesis, q stands for a p-power with $p \in \mathbb{P}$ a prime number.

Definition 1.2.1
The field $\mathbb{F}_q((t))$, with $q = p^l$, is a field of formal power series in the variable t, that is

$$\mathbb{F}_q((t)) = \{ \sum_{n \gg -\infty} a_n t^n \,|\, a_n \in \mathbb{F}_q\}.$$

The field $\mathbb{F}_q((t))$ is a field of formal Laurent series. Especially, the elements of $\mathbb{F}_q((t))$ are the formal series of the form

$$\sum_{i=-m}^{\infty} a_i t^i = a_{-m} t^{-m} + \cdots + a_{-1} t^{-1} + a_0 + a_1 t + \cdots,$$

where $m \in \mathbb{Z}$ and $a_i \in \mathbb{F}_q$.

In order to construct the field $\mathbb{F}_q((t))$ we consider the polynomial ring $\mathbb{F}_q[t]$. The field of fractions of $\mathbb{F}_q[t]$ is the rational function field $\mathbb{F}_q(t)$. Also, we can define a valuation corresponding to the prime element $t \in \mathbb{F}_q[t]$ in the field $\mathbb{F}_q(t)$.

Definition 1.2.2
Let $f \in \mathbb{F}_p(t)$, then we can write

$$f = t^m \tfrac{g}{h}, \textit{ where } g, h \in \mathbb{F}_p[t] \textit{ and } t \nmid gh, (\Leftrightarrow \gcd(gh, t) = 1).$$

Then, we define an exponential valuation υ_t as follows

$$\upsilon_t(f) = m \textit{ and } \upsilon_t(0) = \infty.$$

In addition, we define the associated absolute value $|\cdot|_t$ by

$$|f|_t = p^{-\upsilon_t(f)} = p^{-m}.$$

We can now complete $\mathbb{F}_p(t)$ with respect to υ_t to obtain the field of formal Laurent series over $\mathbb{F}_p$.

An equivalent expression for the above exponential valuation is, for $x = \sum\limits_{i=-m}^{\infty} a_i t^i$

$$\upsilon_t(x) = \upsilon_t\left(\sum_{i=-m}^{\infty} a_i t^i\right) = -m$$

and for the $|\cdot|_t$ we have

$$|x|_t = p^{-\upsilon_t(x)} = p^{-(-m)}.$$

Theorem 1.2.3
The field $\mathbb{F}_p((t))$ is the completion of $\mathbb{F}_p(t)$ with respect to the valuation $|\cdot|_t$.

A detailed proof of the above statement can be found in [17], Theorem 2.20, which is the extended version of [4].

Definition 1.2.4
The formal Laurent series field $\mathbb{F}_q((t))$ equipped with the exponential valuation υ_t is a local field of prime characteristic p, which is called local function field.

We have already mentioned that the formal Laurent series field $\mathbb{F}_q((t))$ is a local field, and especially it is the completion of $\mathbb{F}_q(t)$. Now we will determine its ring of integers, group of units, maximal ideal and residue class field.

The ring of integers of $K := \mathbb{F}_q((t))$ is the ring of formal power series, that is

$$\mathcal{O}_K = \{x \in \mathbb{F}_q((t)) \,|\, \upsilon_t(x) \geq 0\} = \{\sum_{i\geq 0} a_i t^i \,|\, a_i \in \mathbb{F}_q\} = \mathbb{F}_q[[t]]. \tag{1.2}$$

The group of units $\mathcal{U}_K = \{x \in \mathbb{F}_q((t)) \,|\, \upsilon_t(x) = 0\}$ is a subgroup of the formal power series which contains only those elements with a non-zero constant term.

The unique maximal ideal of $\mathcal{O}_K$ is the ring of formal power series which contains those elements whose constant term is zero. This means that,

$$\mathcal{M}_K = \{x \in \mathbb{F}_q((t)) \,|\, \upsilon_t(x) > 0\} = t\mathcal{O}_K$$

Clearly, any uniformizer element of $\mathbb{F}_q((t))$ is a generator of the maximal ideal $\mathcal{M}_K$. Therefore t is a uniformizer element of $\mathbb{F}_p((t))$.

The residue class field of $\mathbb{F}_q((t))$ is the finite field $\mathbb{F}_q$ of q elements, that is

$$\underline{K} = \mathcal{O}_K/\mathcal{M}_K \cong \mathbb{F}_q.$$

K	$\mathbb{Q}_p$	$\mathbb{F}_p((t))$
Completion of	$\mathbb{Q}$	$\mathbb{F}_p(t)$
$\mathcal{O}_K$	$\mathbb{Z}_p$	$\mathbb{F}_p[[t]]$
$\mathcal{M}_K$	$p\mathbb{Z}_p$	$t\mathcal{O}_K$
$\underline{K}$	$\mathbb{F}_p$	$\mathbb{F}_p$

1.3 Basic Galois Theory

Now we will recall some useful concepts. Let K be a local field and L/K be a finite extension of degree n, i.e $[L : K] = n$.

Definition 1.3.1 (Definition 4.1 & 4.7, [32])
Let K be a field. An irreducible polynomial $f \in K[X]$ is separable over K if f has no repeated roots in any splitting field. A polynomial $g \in K[X]$ is separable over K if all irreducible factors of g are separable over K.
Let L/K be an extension of K and let $\alpha \in L$. Then α is separable over K if its irreducible polynomial $\min(\alpha, K)$ is separable over K.
The extension L/K is called separable if every $\alpha \in L$ is separable over K.

Let K be a local field and $f \in K[X]$ is a monic, irreducible and separable polynomial of degree n. Let $\alpha = \alpha^{(1)}, \alpha^{(2)}, \dots, \alpha^{(n)}$ be the roots of $f \in K[X]$ in an algebraic closure $\bar{K}$ of K. The $\alpha^{(i)}$ is called i-th conjugate of α. We can acquire an algebraic extension L/K of K by adjoining a root of f to K, that is

$$L = K(\alpha) \cong K[X]/f(X)K[X].$$

The extension L/K is of degree n and the field L is a vector space over K of dimension n with basis

$$\{1, \alpha, \alpha^2, \dots, \alpha^{n-1}\}.$$

This means that every element $\gamma \in L$ of L can be uniquely written in the form $\gamma = \sum_{i=0}^{n-1} c_i\alpha^i$, with $c_i \in K$ for $1 \leq i \leq n-1$. Thus, the conjugates of γ are $\gamma^{(j)} = \sum_{i=0}^{n-1} c_i(\alpha^{(j)})^i$ for every $1 \leq j \leq n$.

Definition 1.3.2
Using the above notation, the norm of γ is defined as follows: $\mathcal{N}_{L/K}(\gamma) = \prod_{j=1}^{n} \gamma^{(j)}$.

Definition 1.3.3
Let L/K be an algebraic extension and let $f(X) = \sum_{i=0}^{n} a_iX^i \in L[X]$. Then the norm of f is defined to be

$$\mathcal{N}_{L/K}(f(X)) = \prod_{j=1}^{[L:K]} (\sum_{i=0}^{n} a_i^{(j)} X^i),$$

where $a_i^{(j)}$ is the j-th conjugate of a_i.

The coefficients of $\mathcal{N}_{L/K}(f(X))$ are in K, since it is invariant under conjugation.

Definition 1.3.4
Let L/K be a finite algebraic field extension. A K-automorphism σ of L/K is defined to be an automorphism of L that fixes the elements of K. This means that σ is a homomorphism from L to L, $\sigma : L \to L$, such that $\sigma(x) = x$, for every $x \in K$.

*The set of all K-automorphisms is a group under the composition. The group of all K-automorphism of L is called **Galois Group** of L/K and is denoted by $\mathrm{Gal}(L/K)$, that is*

$$\mathrm{Gal}(L/K) = \{\sigma \in \mathrm{Aut}(L) \,|\, \sigma(x) = x, \, \forall x \in K\}.$$

Definition 1.3.5
Let L/K be a finite algebraic field extension of K. The extension L/K is Galois if and only if $[L : K] = |\mathrm{Gal}(L/K)|$.

Definition 1.3.6
Let K be a field and L/K be an algebraic extension of K. Then the following statements are equivalent:

(i.) The extension L/K is Galois.

(ii.) The extension L/K is normal and separable over K.

(iii.) L is the splitting field of a separable polynomial with coefficients in K.

1.4 Extensions of Local Fields

Let K be a field equipped with a discrete valuation v_K and L/K be a finite extension. Then, we can extend the valuation of K to a valuation v_L of L, which is given in the following theorem. The valuation v_L is said to be an extension of the valuation v_K and it will be denoted by $v_L|v_K$.

Theorem 1.4.1 (Theorem (2.5), Chapter II, [9])
Let K be a complete field with respect to a discrete valuation v_K and L/K be a finite extension. Then there is precisely one extension v_L on L of the valuation v_K and

$$v_L = \frac{1}{\mathfrak{f}} v_K \circ \mathcal{N}_{L/K} \text{ with } \mathfrak{f} := \mathfrak{f}(v_L|v_K).$$

The field L is complete with respect to v_L.

Moreover, if L/K is an extension of local fields and v_K is a discrete valuation of K, then the extended valuation v_L on L is again discrete.

Let K be a local field, L/K be a finite extension of K and v_L is the unique discrete exponential valuation of L, which is the extension of the discrete exponential valuation v_K of K. If $\mathcal{O}_K$, $\mathcal{M}_K$, π_K are the ring of integers, the unique maximal ideal and the uniformizing element of K, respectively, and $\mathcal{O}_L$, $\mathcal{M}_L$, π_L are the ring of integers, the unique maximal ideal and the uniformizing element of L, respectively. Then, the ring of integers $\mathcal{O}_K$ of K is a subring of the ring of integers $\mathcal{O}_L$ of L and the maximal ideal $\mathcal{M}_K$ coincides with $\mathcal{M}_L \cap \mathcal{O}_K$, that is

$$\mathcal{M}_K = \mathcal{M}_L \cap \mathcal{O}_K.$$

Also, we obtain the inclusions $\underline{K} \subseteq \underline{L}$ and $v_K(K^\times) \subseteq v_L(L^\times)$.

$$\begin{tikzcd}
L \arrow[d, dash] & \mathcal{O}_L \arrow[d, dash] & \mathcal{M}_L \arrow[d, dash] & \underline{L} \arrow[d, dash] \\
K & \mathcal{O}_K & \mathcal{M}_K & \underline{K}
\end{tikzcd}$$

The index

$$e = e(v_L|v_K) = (v_L(L^\times) : v_K(K^\times))$$

is called the ***ramification index*** $e(L/K)$ of the extension L/K.

The degree

$$\mathfrak{f} = \mathfrak{f}(v_L|v_K) = [\underline{L} : \underline{K}]$$

is called the ***inertia degree*** $\mathfrak{f}(L/K)$ of the extension L/K.

In addition, if $x \in K$, then $v_L(x) = e(v_L|v_K)v_K(x)$.

Since v_L, v_K are discrete, we have that the inclusion $v_K(K^\times) \subseteq v_L(L^\times)$ is equivalent to $v_K(\pi_K)\mathbb{Z} \subseteq v_L(\pi_L)\mathbb{Z}$ such that $v_K(\pi_K) = ev_L(\pi_L)$, which yields

$$\pi_K = \varepsilon\pi_L^e, \text{ where } \varepsilon \in \mathcal{U}_L.$$

Additionally, we obtain the following result $\mathcal{M}_K\mathcal{O}_L = \pi_K\mathcal{O}_L = \pi_L^e\mathcal{O}_L = \mathcal{M}_L^e$ and so

$$\mathcal{M}_K = \mathcal{M}_L^e.$$

Proposition 1.4.2 (Proposition (6.8), [35])
One has $[L : K] \geq e\mathfrak{f}$ and the fundamental identity $[L : K] = e\mathfrak{f}$, if v_K is discrete and L/K is separable.

Since, in our case, v_K is a discrete exponential valuation and the extension L/K is separable, then it holds the fundamental identity: $[L : K] = e\mathfrak{f}$, and especially it holds that $[L : K] = n = e\mathfrak{f}$.

Definition 1.4.3
The extension L/K is called ***unramified*** *if the extension of residue class fields $\underline{L}/\underline{K}$ is separable and $[\underline{L} : \underline{K}] = [L : K]$. Equivalently, the extension is unramified if $e = 1$, (since in our case $[L : K] = ef$ and $\underline{L}/\underline{K}$ is separable).*

The extension L/K is called ***totally ramified*** *if $e = n$, (equivalently $\mathfrak{f}(v_L|v_K) = 1$).*
The extension L/K is called ***tamely ramified*** *if the extension of residue class fields $\underline{L}/\underline{K}$ is separable and $p \nmid e$ ($\Leftrightarrow \gcd(e, p) = 1$) where the prime $p > 0$ is the characteristic of the residue field of K.*
The extension L/K is called ***wildly ramified*** *if the extension of residue class fields $\underline{L}/\underline{K}$ is separable and $p \mid e$ where the prime $p > 0$ is the characteristic of the residue field of K.*

Let L/K be an algebraic extension of the local field K. Then we can split the extension L/K into a tower of extensions

$$K \subseteq K^{\mathrm{ur}} \subseteq L,$$

where L/K^{ur} is totally ramified and K^{ur}/K is unramified extension.

$$[L : K] = e\mathfrak{f} \left\{ \begin{tikzcd} L \arrow[d, dash, "\begin{array}{l}\text{Totally ramified of degree}\\ {[L : K^{\mathrm{ur}}] = e(L|K)}\end{array}"] \\ K^{\mathrm{ur}} \arrow[d, dash, "\begin{array}{l}\text{Unramified of degree}\\ {[K^{\mathrm{ur}} : K] = \mathfrak{f}(L|K)}\end{array}"] \\ K \end{tikzcd} \right.$$

We will examine the above types of extensions of $\mathbb{F}_q((t))$ in the next sections. For more details we refer the reader to Sections 2 and 3, Chapter II in [9] and Section 7, Chapter II in [35].

Let $\bar{K}$ be the an algebraic closure of K. To simplify notation, the unique extension of v_K to $\bar{K}$ is also denoted by v_K.

Definition 1.4.4
For $\gamma, \gamma' \in \bar{K}$ we write $\gamma \sim \gamma'$ if $v_K(\gamma - \gamma') > v_K(\gamma)$.

1.5 Unramified Extensions

We start by describing the case of the unramified extensions. Several results in this case carried over from the study of the unramified extensions of p-adic fields.

Let L_1/K and L_2/K be two extensions of K, and L_1, L_2 contained in some common extension L/K. The composite of L_1 and L_2 is denoted by L_1L_2 and is defined to be the subfield of L generated by L_1 and L_2; that is $L_1L_2 = L_1(L_2) = L_2(L_1)$. Equivalently, this means that the composite is the smallest field containing both L_1 and L_2. It can be described by adjoining the generators of K to L, or alternatively, adjoining the generators of L to K.

It can be proved that the composite of all finite unramified extensions of K in a fixed algebraic closure $\bar{K}$ of K is unramified. The composite of all finite unramified extensions of K is called ***maximal unramified extension*** K^{ur} of K. The maximality of this extension implies that

$$\sigma(K^{\mathrm{ur}}) = K^{\mathrm{ur}},$$

for every K-automorphism σ of the separable closure K^{sep} over K. Therefore, the extension K^{ur}/K is Galois.

One of the most important results of p-adic fields is Hensel's Lemma (see Theorem 4.1, [33] or Theorem 4.5.2, [14]), and this is also true for local function fields of the form $\mathbb{F}_p((t))$.

Theorem 1.5.1 (Hensel's Lemma, Theorem 4.1 [17])
Let K be a field, $f \in \mathcal{O}_K[X]$ be a polynomial in $\mathcal{O}_K[X]$ and $\alpha_0 \in \mathcal{O}_K$ such that

$$f(\alpha_0) \equiv 0 \mod \mathcal{M}_K \text{ and } f'(\alpha_0) \not\equiv 0 \mod \mathcal{M}_K.$$

Then, there exists $\alpha \in \mathcal{O}_K$ such that α is a root of f, that is $f(\alpha) = 0$ and $\alpha \equiv \alpha_0 \mod \mathcal{M}_K$.

Another important statement in the case of unramified extensions is the fact that for every positive integer $\mathfrak{f}$, we can show that there exists a unique unramified extension of K of degree $\mathfrak{f}$. We can obtain this extension by adjoining a primitive $(p^{\mathfrak{f}}-1)$-th root of unity to $K=\mathbb{F}_p((t))$.

Theorem 1.5.2 (Theorem 4.4 & 4.5, [17])
For every integer $\mathfrak{f}\geq 1$ there exists a unique unramified extension of the field $K=\mathbb{F}_p((t))$ of degree $\mathfrak{f}$. The unique unramified extension K^{ur}/K of degree $\mathfrak{f}$ of $K=\mathbb{F}_p((t))$ is obtained by adjoining a primitive $(p^{\mathfrak{f}}-1)$-th root of unity to K. Consequently, every unramified extension is a Galois extension.

Proposition 1.5.3 (Proposition (1.2), [9])
The maximal unramified extension K^{ur} of $K=\mathbb{F}_q((t))$, where $q=p^{\mathfrak{f}}$, is a Galois extension. The Galois group is isomorphic to $\hat{\mathbb{Z}}$ and topologically generated by an automorphism φ_K, such that

$$\varphi_K(\alpha)\equiv\alpha^q \mod \mathcal{M}_{K^{\mathrm{ur}}} \quad \textit{for } \alpha\in\mathcal{O}_{K^{\mathrm{ur}}}.$$

Proposition 1.5.4
Let K be a local field and $f\in K[X]$. If $\underline{f}\in\underline{K}[X]$ is square free, then the unramified extension of K of degree

$$\operatorname{lcm}\{\deg(f_i)\ \mid\ f_i \textit{ is an irreducible factor of } \underline{f}\}$$

is the splitting field of f.

1.6 Totally Ramified Extensions

Let L/K be a finite extension of the field K and let the subfield K^{ur} of L be the maximal unramified extension of K. Then we have that the extension L/K^{ur} is totally ramified. Therefore, we have split the extension L/K into two extensions. This means that we are able to construct all the finite extensions of K by examining their unramified and totally ramified extensions.

Definition 1.6.1
Let $f(X)\in\mathcal{O}_K[X]$ be a monic polynomial of the form

$$f(X)=X^n+a_{n-1}X^{n-1}+\cdots+a_0.$$

*The polynomial f is called **Eisenstein polynomial** if $v_K(a_i)\geq 1$, for every $1\leq i\leq n-1$ and $v_K(a_0)=1$.*

An equivalent definition of the Eisenstein polynomial can be described in terms of the maximal ideal. Specifically, the polynomial $f(X)\in\mathcal{O}_K[X]$ is said to be Eisenstein if

$$a_0,\dots,a_{n-1}\in\mathcal{M}_K \text{ and } a_0\notin\mathcal{M}_K^2.$$

Theorem 1.6.2
Let K be a non-Archimedean local field of prime characteristic p and let L/K be a finite extension of degree n. Let also π_L be a prime element of L. The extension L/K is totally ramified if and only if π_L is a primitive element of L/K, that is $L=K(\pi_L)$, and π_L is a root of an Eisenstein polynomial of degree n.

For more details we refer the reader to Section 2, Chapter 14 in [18] or Proposition (3.6), Chapter II, p. 54 in [9].

It is worth pointing out that if K^{ur} is the maximal unramified subextension of L/K, then we know that L/K^{ur} is totally ramified. Thus, according to Theorem 1.6.2, we have that $L=K^{\mathrm{ur}}(\pi_L)$, where π_L is a prime element of L and satisfies an Eisenstein polynomial of degree $[L:K^{\mathrm{ur}}]$ with coefficients in K^{ur}.

Moreover, by the above theorem, we obtain that every Eisenstein polynomial over a discrete valued field is irreducible.

1.7 Tamely Ramified Extensions

Let L/K be an algebraic extension inside the algebraic closure $\bar{K}/K$. It holds that every subextension of a tamely ramified extension is tamely ramified. Also, the composite of tamely ramified extensions inside the algebraic closure $\bar{K}/K$ is tamely ramified. For more details we refer the reader to Corollaries (7.8) and (7.9), Section 7, Chapter II in [35].

Definition 1.7.1
*Let L/K be an algebraic extension. Then the composite of all tamely ramified subextensions of L/K inside the algebraic closure $\bar{K}/K$ is called **maximal tamely ramified** subextension of L/K. We denote K^{tr} the composite of all tamely ramified subextensions of L/K.*

In general, we have obtained the following situation

$$K \subseteq K^{\mathrm{ur}} \subseteq K^{\mathrm{tr}} \subseteq L.$$

If L/K is finite and $e = e(L/K) = e'p^k$, where $\gcd(p, e') = 1$, then the extension $K^{\mathrm{tr}}/K^{\mathrm{ur}}$ has degree $[K^{\mathrm{tr}}/K^{\mathrm{ur}}] = e'$.

Let K be a local function field with ring of integers $\mathcal{O}_K$, unique maximal ideal $\mathcal{M}_K$ and residue class field $\underline{K}$. Let also $f(X) = a_n X^n + \cdots + a_0 \in \mathcal{O}_K[X]$. Then, we will denote

$$\underline{f}(X) = \underline{a}_n X^n + \cdots + \underline{a}_0 \in \underline{K}[X].$$

In addition, if $f(X) - g(X) \in \mathcal{M}_K^l[X]$, then we will write

$$f(X) \equiv g(X) \mod \mathcal{M}_K^l.$$

Lemma 1.7.2 (Abhyankar's Lemma, [34], Chapter 5, Section 2)
If L/K is a tamely ramified extension and M/K is a finite extension, and additionally $e(L/K) \mid e(M/K)$, then LM/M is unramified.

1.8 Newton Polygons

In this section we study the Newton polygon of a polynomial. For a deeper discussion of the latter we refer the reader to Section 6, Chapter II in [35] or Chapter 9 in [33].

Hensel's Lemma, (Theorem 1.5.1), is a useful tool that gives us a method of finding roots of a polynomial over the ring of integers $\mathcal{O}_K$ of a field K. However, we would like to find a method for the determination of the roots of a polynomial in $\bar{K}$. Moreover, the Newton polygon of any polynomial f over K gives us information for the factors of f.

Definition 1.8.1
Let v be an exponential valuation of the field K and let $f(X) = a_n X^n + \cdots + a_1 X + a_0 \in K[X]$ be a polynomial over K such that $a_0 a_n \neq 0$. Consider the set of points

$$\{(i, v(a_i)) \in \mathbb{R}^2 \,:\, 0 \leq i \leq n\}.$$

The Newton polygon of f is defined as the "lower convex hull" of that set of points. Also, the Newton polygon is a polygonal chain which consists of a sequence of segments with increasing slopes.

The points where the slopes change, say $(i_j, v(a_{i_j}))$ are called the vertices of the Newton polygon.

Let $(x_0, y_0) = (0, v(a_0)), (x_1, y_1), \dots, (x_s, y_s)$ denote these vertices, then the numbers $\frac{y_i - y_{i-1}}{x_i - x_{i-1}}$ are called slopes of f.

Comment 1.8.2
If $a_i = 0$, we have $v(a_i) = \infty$, and we omit that point.

Now we describe more precisely the Newton polygon construction of a polynomial. The Newton polygon is constructed by taking the vertical line through $(0, v(a_0))$ and rotating it counterclockwise, with $(0, v(a_0))$ as a center, until it hits any of the points $(i, v(a_i))$, say $(i_1, v(a_{i_1}))$. Then, we take the new point $(i_1, v(a_{i_1}))$ as the center and rotating the line further until it hits another one of the points $(i, v(a_i))$, say $(i_2, v(a_{i_2}))$. We repeat this process until we reach the point $(n, v(a_n))$.

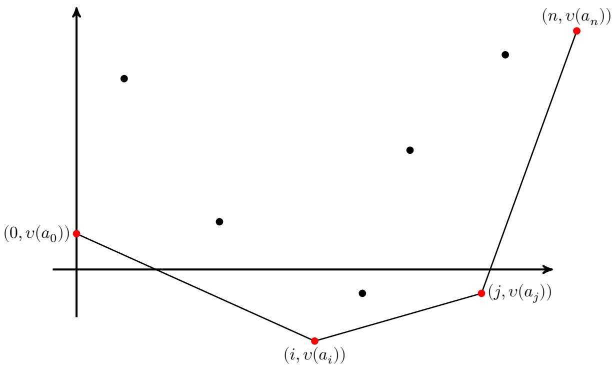


Figure 1.1: **Newton Polygon**

Newton polygons can be used to investigate the valuations of the roots of a polynomial.

Proposition 1.8.3 (Proposition (6.3), Chapter II, [35])
Let $f(X) = a_nX^n + \cdots + a_1X + a_0 \in K[X]$ be a polynomial over a complete field K such that $a_0a_n \neq 0$, v be an exponential valuation of the field K and w be the extension of v to the splitting field L of f.

If the line segment $(r, v(a_r)) \longleftrightarrow (s, v(a_s))$ of slope $-m$ occurs in the Newton polygon of f, then f has exactly $s - r$ roots

$$\{\alpha_1, \dots, \alpha_{s-r}\}$$

with valuations $w(\alpha_1) = w(\alpha_2) = \cdots = w(\alpha_{s-r}) = m$.

Moreover, we can use the Newton polygon in order to investigate the way of the factorization of a polynomial. It can be proved that given a polynomial over a field K, the slopes of its Newton polygon corresponds to its factorization in K.

Theorem 1.8.4 (Proposition (6.4), Chapter II in [35])
Let f be a polynomial over K whose Newton polygon has slopes $-m_r < \cdots < -m_1$. If the valuation v has a unique extension w to the splitting field L of f, then we obtain the factorization

$$f(X) = a_n \prod_{j=1}^{r} f_j(X),$$

where each factor f_j is a polynomial over K as follows:

$$f_j(X) = \prod_{w(\alpha_i)=m_j} (X - \alpha_i) \in K[X],$$

but f_j is not irreducible.

According to Theorem 1.8.4, we have that the Newton polygon is a powerful tool. Once we have computed the Newton polygon of any given polynomial f, we have some information of its factorization. In particular, if its Newton polygon contains r segments, the polynomial f will factor into r factors, which may or may not themselves be irreducible.

There are many interesting consequences which are arising from Theorem 1.8.4. Firstly, by the above theorem, we observe that if $f \in K[X]$ is irreducible, then all the roots of f have the same valuation. This yields that the Newton polygon of $f \in K[X]$ consists of exactly one segment with slope of length equal to the degree of f. Particularly, if the Newton polygon consists of more than one segments, then necessarily the polynomial is reducible in $K[X]$. However, the converse is not true. We can prove that there exists a polynomial in $K[X]$ which is reducible but whose Newton polygon consists of exactly one segment.

Another interesting consequence is the Eisenstein irreducibility criterion. We notice that if the Newton polygon of a polynomial consists of exactly one segment joining $(0, d)$ and (n, m) with $m - d$ coprime to n, then the polynomial is irreducible. Indeed, in this situation we have that any root α of f has valuation $v_K(\alpha) = \frac{m-d}{n}$. If f factored as a product of two polynomials of smaller degree, then at least one root α of f would have valuation equal to a fraction with denominator strictly smaller than n. But this is a contradiction to Theorem 1.8.4. In particular, if f is Eisenstein, then its Newton polygon consists of exactly one segment joining $(0, 1)$ and $(n, 0)$ with slope

$\frac{-1}{n}$. Thus, by the above discussion, we have that it is irreducible.

It is worth bearing in mind that the Eisenstein irreducibility criterion is a special case of the Irreducibility Criterion which is given in the next proposition.

Proposition 1.8.5
If the Newton polygon of $f \in K[X]$ consists of only one segment with no integer points apart from the endpoints, then f is irreducible.

Every Eisenstein polynomial is irreducible, and thus it can be used to generate a local field extension. In order to determine the type of extension, we consider another important, well-known result regarding Newton polygons.

We note that from now on $\mathcal{NP}(f)$ stands for the Newton polygon of a polynomial $f \in K[X]$.

Proposition 1.8.6
Let $\mathcal{NP}(g)$ denote the Newton polygon for some $g \in \mathcal{O}_K[X]$. If the slopes of the segments of $\mathcal{NP}(g)$ are in lowest terms, then their denominators divide the ramification indices of the extensions defined by the irreducible factors of g.

1.9 Splitting Fields and Galois Groups of Tamely Ramified Extensions

Tamely ramified extensions can be described very nicely theoretically. They always consist of a radical extension over the (cyclic) maximal unramified relative extension. This is why one can easily determine the generating automorphisms of the Galois group for Galois extensions. Helmut Hasse studies this thoroughly in Chapter 16 of his book [18]. Based on these results, we present some results about tamely ramified subfields and we give the splitting fields and the Galois groups of tamely ramified extensions.

Proposition 1.9.1 (Proposition 2.1, [16])
Let $n = e_0 p^w$ with $p \nmid e_0$ and let $f(X) = X^n + \sum_{i=1}^{n-1} a_i X^i + a_0 \in \mathcal{O}_K[X]$ be a polynomial whose Newton polygon is a line of slope $-h/n$, where $\gcd(h, n) = 1$. Let α be a root of f. The maximal tamely ramified subextension M of $L := K(\alpha)$ of degree e_0 can be generated by the Eisenstein polynomial

$$X^{e_0} - (-\psi_0)^b \pi^{e_0 a}$$

with $\psi_0 \equiv a_0 \mod \pi^{h+1}$ and where a and b are integers such that $ae_0 + bh = 1$.

This proposition is a useful tool, because it informs us that every totally and tamely ramified extension of degree e can be generated by a polynomial of the form $X^e - \gamma\pi_K$ where $v_K(\gamma) = 0$.

Corollary 1.9.2 (Corollary 2.2, [16])
Let $f(X) = \sum_{i=0}^{e} a_i X^i \in \mathcal{O}_K[X]$ be an Eisenstein polynomial and assume $p \nmid e$. If $\psi(X) = X^e + \psi_0$ with $\psi_0 \equiv a_0 \mod \langle \pi^2 \rangle$, then the extensions generated by f and ψ are isomorphic.

Let $f(X) = \sum_{i=0}^{e} a_i X^i \in \mathcal{O}_K[X]$ be an Eisenstein polynomial with $p \nmid e$. If ζ_e denotes a primitive e-th root of unity, then Corollary 1.9.2 yields that the splitting field of f is

$$N := \mathrm{Spl}(f) = K(\zeta_e, \sqrt[e]{-a_0}).$$

The Galois group of N/K is well known. We obtain it by the general description of Galois groups of normal, tamely ramified extensions (see [18], Chapter 16 for more details).

Theorem 1.9.3 (Theorem 2.3, [16])
Let K be a local field, and let q be the size of its residue class field. Let N/K be a normal, tamely ramified extension with ramification index e and inertia degree $\mathfrak{f}$. There exists an integer r with

$r(q-1) \equiv 0 \mod e$ such that $N = K(\zeta, \sqrt[e]{\zeta^r \pi})$, where ζ is a primitive $(q^{\mathfrak{f}}-1)$-st root of unity and $q^{\mathfrak{f}} - 1 \equiv 0 \mod e$. Let $\kappa := \frac{r(q-1)}{e}$. The generators of the Galois group are the automorphisms

$$\sigma : \zeta \mapsto \zeta, \ \sqrt[e]{\zeta^r \pi} \mapsto \zeta^{(q^{\mathfrak{f}}-1)/e} \sqrt[e]{\zeta^r \pi} \ \text{and} \ \tau : \zeta \mapsto \zeta^q, \ \sqrt[e]{\zeta^r \pi} \mapsto \zeta^{\kappa} \sqrt[e]{\zeta^r \pi}.$$

The Galois group of N/K as a finitely generated group is

$$\mathrm{Gal}(N/K) \cong \langle \sigma, \tau | \sigma^e = 1, \tau^{\mathfrak{f}} = \sigma^r, \sigma\tau = \tau\sigma^q \rangle.$$

It should be noted that more information for the above Theorem 1.9.3 can be found in Chapter 16 of [18]. We can briefly describe some of the main parts analyzed in that chapter in the following theorem. We give it without a proof.

Theorem 1.9.4
Let K be a local field with $\underline{K} = \mathbb{F}_q$ its residue class field. Let L/K be a tamely ramified extension with ramification index e and inertia degree $\mathfrak{f}$. Then:

i. The extension L/K is isomorphic to one of the extensions of the form

$$K(\zeta, \sqrt[e]{\zeta^r \pi}), \quad \text{with} \quad r \in \{0, \dots, d_e^{(\mathfrak{f})} - 1\}$$

where ζ is a primitive $(q^{\mathfrak{f}} - 1)$-st root of unity and $d_e^{(\mathfrak{f})} := \gcd(e, q^{\mathfrak{f}} - 1)$.

ii. In particular, two extensions $K(\zeta, \sqrt[e]{\zeta^r \pi})$ and $K(\zeta, \sqrt[e]{\zeta^{r'} \pi})$ are conjugate if and only if $r' \equiv r \mod d_e^{(\mathfrak{f})}$.

iii. The extension L/K is Galois if and only if $e \mid q^{\mathfrak{f}} - 1$ and $e \mid r(q-1)$. Then, the generators of the Galois group are the automorphisms

$$\sigma : \zeta \mapsto \zeta, \ \sqrt[e]{\zeta^r \pi} \mapsto \zeta^l \sqrt[e]{\zeta^r \pi} \ \text{and} \ \tau : \zeta \mapsto \zeta^q, \ \sqrt[e]{\zeta^r \pi} \mapsto \zeta^{\kappa} \sqrt[e]{\zeta^r \pi},$$

where $\kappa := \frac{r(q-1)}{e}$ and $l := \frac{q^{\mathfrak{f}}-1}{e}$. The Galois group of L/K as a finitely generated group is

$$\mathrm{Gal}(L/K) \cong \langle \sigma, \tau | \sigma^e = 1, \tau^{\mathfrak{f}} = \sigma^r, \sigma\tau = \tau\sigma^q \rangle.$$

Proof. For more details see [18], Chapter 16. □

Remark 1.9.5
It is worth pointing out that in the part (i.) of Theorem 1.9.4 the condition $r \in \{0, \dots, d_e^{(\mathfrak{f})} - 1\}$ means that we consider $r \in \mathbb{Z}/d_e^{\mathfrak{f}}\mathbb{Z}$.

According to Corollary 1.9.2, we have that in the case of a tamely ramified extensions, the last coefficient of the corresponding Eisenstein polynomial directly provides the representation as a radical extension. From this representation it is now easy to get the parameter r described above.

In case that a tamely ramified extension L/K is not normal, we can obtain a similar statement. We can compute its normal closure by increasing the inertia degree. (This means that only unramified parts can be added in order to determine its normal closure.) This results in a Galois group with a similar presentation. Compare to [15, Satz 3.6], and [20, Proposition 3.5.1].

Theorem 1.9.6
Let ζ be a primitive $(q^{\mathfrak{f}} - 1)$-st root of unity and let $L = K(\zeta, \sqrt[e]{\zeta^r \pi})$ be tamely ramified. Let $g = \gcd(q^{\mathfrak{f}} - 1, r(q-1))$, and let $u \in \mathbb{N}$ be minimal such that

$$q^{\mathfrak{f}u} - 1 \equiv 0 \mod (e(q^{\mathfrak{f}} - 1)/g).$$

Let ξ be a primitive $(q^{\mathfrak{f}u} - 1)$-st root of unity, and let $s = r(q^{\mathfrak{f}u} - 1)/(q^{\mathfrak{f}} - 1)$. Then

$$N = K(\xi, \sqrt[e]{\xi^s \pi})$$

is the normal closure of L/K, and the Galois group of L/K is

$$\mathrm{Gal}(N/K) \cong \langle x, y | x^e = 1, y^{\mathfrak{f}u} = x^s, xy = yx^q \rangle.$$

In Theorems 1.9.3 and 1.9.6, the third relation in the Galois group is equivalent to one group generator acting on the other through conjugation. This action is the same as raising one of the generators to a power q that is coprime to its order e. It is tempting to believe that the Galois group of a tamely ramified extension is the semidirect product of nontrivial cyclic groups. However, this is not always true, as our next example demonstrates.

Example 1.9.7
Let ζ_8 be a primitive eighth root of unity. Let us also denote $K := \mathbb{F}_3((t))$. We consider the local function field $L = K(\zeta, \sqrt[4]{\zeta^2 \cdot t})$. Additionally, one can prove that L is the field which is generated by the polynomial

$$f(X) = X^8 + t^2X^4 + t^2 \in K[X]$$

over K. Then one can find that the extension L/K has ramification index $e = e(L/K) = 4$ and inertia degree $\mathfrak{f} = \mathfrak{f}(L/K) = 2$. Furthermore, by construction we have that $r = 2$ and the number of elements in the residue class field $\underline{K}$ of K is $q = 3$.

Then computing the Galois group of L/K we check whether the extension is normal so as to apply the above theorems. We can verify that L/K is normal since $e \mid (q^{\mathfrak{f}} - 1) \Leftrightarrow 4 \mid 3^2 - 1$ and $e \mid r(q-1) \Leftrightarrow 4 \mid 2(3-1)$. Therefore, according to Theorem 1.9.3 we have that

$$\mathrm{Gal}(L/K) \cong \langle \sigma, \tau | \sigma^e = 1, \tau^{\mathfrak{f}} = \sigma^r, \sigma\tau = \tau\sigma^q \rangle = \langle \sigma, \tau | \sigma^4 = 1, \tau^2 = \sigma^2, \sigma\tau = \tau\sigma^3 \rangle.$$

However, we know that the Quaternion group of 8 elements is of the form

$$Q_8 \cong \langle a, b \,|\, a^4 = 1,\, a^2 = b^2,\, b^{-1}ab = a^{-1} \rangle.$$

In the above representation of Q_8 we get that $a^{-1} = a^3$, hence, it is clear that

$$\mathrm{Gal}(L/K) \cong Q_8.$$

Nevertheless, it is important to recall that the Quaternion group Q_8 is no a semidirect product of cyclic groups.

1.9.1 Computation of Galois Groups

Considering all the above we can give in algorithmic form the computation of the splitting field and Galois group in the case of tamely ramified extensions. We begin with the Galois group algorithm. For more details see Algorithm 3.1 in [15].

Algorithm 1: Galois Group of Tamely Ramified Extension [TameGaloisGroup(f)]

```
Input: An irredicible polynomial $f \in K[X]$, where $K := \mathbb{F}_q((t))$, of degree $n$, which generates a tamely
       ramified extension $L/K$.
Output: The Galois group of $f$ as a finitely generated group.
Determine the inertia degree $\mathfrak{f}$ of the subextension $U/K$ and the ramification index $e$ as well as an
  Eisenstein polynomial $\sum_{i=0}^{e} a_i X^i \in U[X]$ of the ramified extension $L/U$;
Define $d_e^{(\mathfrak{f})} \leftarrow \gcd(e, q^{\mathfrak{f}} - 1)$ and select $\zeta \in U$ to be a primitive $(q^{\mathfrak{f}} - 1)$-th root of unity;
Determine $r \in \{0, \dots, d_e^{(\mathfrak{f})} - 1\}$ with $r \equiv r' \mod d_e^{(\mathfrak{f})}$, where $r'$ such that $-a_0/\pi \equiv \zeta^{r'} \mod \pi\mathcal{O}_U$;
Define $g \leftarrow \gcd(q^{\mathfrak{f}} - 1, r(q-1))$ and initialize $u \leftarrow 1$;
while $q^{\mathfrak{f}u} - 1 \not\equiv 0 \mod e \cdot \frac{q^{\mathfrak{f}}-1}{g}$ do
    Set $u \leftarrow u + 1$;
Determine $q \leftarrow q \mod e$, and $r \leftarrow r \cdot \frac{q^{\mathfrak{f}u}-1}{q^{\mathfrak{f}}-1} \mod d_e^{(\mathfrak{f}u)}$, where $d_e^{(\mathfrak{f}u)} = \gcd(e, q^{\mathfrak{f}u} - 1)$;
return The Galois group of $f$;   $\mathrm{Gal}(L/K) \cong \langle \sigma, \tau | \sigma^e = 1, \tau^{\mathfrak{f}} = \sigma^r, \sigma\tau = \tau\sigma^q \rangle$
```

Next the spitting field algorithm is given.

Algorithm 2: Splitting Field of Tamely Ramified Extension [TameSplFldLFF(f)]

```
Input: An irredicible polynomial $f \in K[X]$, where $K := \mathbb{F}_q((t))$, of degree $n$, which generates a tamely
       ramified extension $L/K$.
Output: The splitting field of $f$ in the form $\mathbb{F}_{q^\ell}((u_\ell))$.
Determine the inertia degree $\mathfrak{f}$ of the subextension $U/K$ and the ramification index $e$ as well as an
  Eisenstein polynomial $\sum_{i=0}^{e} a_i X^i \in U[X]$ of the ramified extension $L/U$;
Define $d_e^{(\mathfrak{f})} = \gcd(e, q^{\mathfrak{f}} - 1)$ and select $\zeta \in U$ to be a primitive $(q^{\mathfrak{f}} - 1)$-th root of unity;
Determine $r \in \{0, \dots, d_e^{(\mathfrak{f})} - 1\}$ with $r \equiv r' \mod d_e^{(\mathfrak{f})}$, where $r'$ such that $-a_0/\pi \equiv \zeta^{r'} \mod \pi\mathcal{O}_U$;
Define $g \leftarrow \gcd(q^{\mathfrak{f}} - 1, r(q-1))$ and initialize $u \leftarrow 1$;
while $q^{\mathfrak{f}u} - 1 \not\equiv 0 \mod e \cdot \frac{q^{\mathfrak{f}}-1}{g}$ do
    Set $u \leftarrow u + 1$;
Determine $q \leftarrow q \mod e$, and $r \leftarrow r \cdot \frac{q^{\mathfrak{f}u}-1}{q^{\mathfrak{f}}-1} \mod d_e^{(\mathfrak{f}u)}$, where $d_e^{(\mathfrak{f}u)} = \gcd(e, q^{\mathfrak{f}u} - 1)$;
Select $\xi$ to be a primitive $(q^{\mathfrak{f}u} - 1)$-th root of unity;
return the splitting field
    $L = K(\xi, \sqrt[e]{\xi^r \pi}) \cong \mathbb{F}_{q^{\mathfrak{f}u}}((u_\ell))$
  of $f$;
```

1.10 Ramification Groups

The ramification groups define a sequence of decreasing normal subgroups of the Galois group which are eventually trivial. The ramification groups give structural information about the Galois group. For a deeper discussion of them, we refer the reader to Section 10, Chapter II in [35] and the relevant section of Chapter 7 in [28].

Definition 1.10.1
Let L/K be a Galois extension with Galois group $G := \mathrm{Gal}(L/K)$ and v_L is the discrete valuation of L. For an integer $i \geq -1$, the i-th ramification group of G is defined to be

$$G_i := \{\sigma \in G \mid v_L(\sigma(x) - x) \geq i + 1 \,\text{for all}\, x \in \mathcal{O}_L\}.$$

It is clear that $G_{-1} = G$. The group G_0 is called the inertia subgroup of G and the group G_1 is called the ramification subgroup of G. For $i > 1$, the subgroups G_i are called the higher ramification groups of the extension L/K. Each group G_i is a normal subgroup of G and satisfies $G_i \trianglelefteq G_j$ whenever $i > j$. Also, G_i is trivial for sufficient large values of i.

Moreover, it can be proved that given a Galois extension L/K, the group G_0 is trivial if and only if the extension L/K is unramified. Additionally, the group G_1 is trivial if and only if the extension L/K is tamely ramified.

The ramification groups form the chain

$$G := G_{-1} \trianglerighteq G_0 \trianglerighteq G_1 \trianglerighteq \dots \trianglerighteq G_k = \{\mathrm{id}\}.$$

Therefore, the ramification groups of G form a filtration of G. If the consecutive groups are not equal, that is $G_i \neq G_{i+1}$, then we would like to determine the values of the index i such that $G_i \neq G_{i+1}$, since they give us interesting results.

Definition 1.10.2
Let L/K be a finite Galois extension with Galois group G. Integers i such that $G_i \neq G_{i+1}$ are called ramification breaks of L/K.

Some interesting properties of the ramification breaks are given below.

Proposition 1.10.3
If G is an abelian group, then every ramification break must be divisible by the order of G_0/G_1.

Proposition 1.10.4 (Proposition 4.5 [9])
Let p be the characteristic of the residue class field $\underline{L}$ and i, j be any two ramification breaks of L/K. Then $i \equiv j \mod p$.

Since the ramification groups are subgroups of $\mathrm{Gal}(L/K)$, they correspond to subfields of L/K. In the cases of G_0 and G_1 these subfields are well known.

Proposition 1.10.5
Let L/K be Galois extension with Galois group G and G_i be the i-th ramification group of G. Then,

1. *The maximal unramified subfield K^{ur} of L/K is the fixed field of the inertia group G_0, Therefore, $G_0 := \mathrm{Gal}(L/K^{\mathrm{ur}})$. In addition, G_0 is a normal subgroup of order $e_{L/K}$ with cyclic quotient of order $\mathfrak{f}_{L/K}$.*

2. *The maximal tamely ramified subfield K^{tr} of L/K is the fixed field of the first ramification subgroup G_1, Therefore, $G_1 := \mathrm{Gal}(L/K^{\mathrm{tr}})$.*

Chapter 2

Ramification Polygon & Applications

This chapter is devoted to the study of two invariants -the ramification polygon of an Eisenstein polynomial and the residual polynomial of an extension. The material studied for the case of p-adic fields, i.e $\mathbb{Q}_p$, can be found in [15], [16] and [29]. In our case, we study them for the case of local function fields of prime characteristic, i.e. $\mathbb{F}_q((t))$. Moreover, for the sake of completeness the results and theory of the aforementioned invariants are summarized in algorithmic forms. Those algorithms are foundational to our approach to compute the splitting field and Galois group of a given polynomial over $\mathbb{F}_q((t))$.

Throughout the thesis, we write $K := \mathbb{F}_q((t))$ for a local function field over the finite field $\mathbb{F}_q$, with q a p-power. Also, $v_K = v_t$ stands for the valuation of K, as defined in Chapter 1.

2.1 Ramification Polygons

A useful tool for obtaining information about the splitting field and the Galois group of a polynomial is the ramification polygon. We can further study a totally ramified extension from its ramification polygon. Ramification polygons were first used by Krasner in [26], in which he developed the ramification theory of non-normal p-adic fields. Also, they have been used to study ramification groups and reciprocity in [44], to compute splitting fields and Galois groups in [16] and to classify extensions in [31].

Definition 2.1.1
Let $f(X) := a_nX^n + \cdots + a_1X + a_0 \in \mathcal{O}_K[X]$ be an Eisenstein polynomial which defines the extension L/K, α be a root of f and set $L := K(\alpha)$. Then, the polynomial

$$\rho_f(X) := \sum_{i=0}^{n} \rho_i X^i = \frac{f(\alpha X + \alpha)}{\alpha^n} \in \mathcal{O}_L[X]$$

is called ramification polynomial of f. The Newton polygon of the ramification polynomial is called ramification polygon, denoted by $\mathcal{RP}(f)$.

If we denote the roots of f in some algebraic closure by $\alpha = \alpha_1, \cdots, \alpha_n$, then we have that

$$\rho_f(X) = X\prod_{i=2}^{n}(X - \frac{\alpha_i - \alpha}{\alpha}) = X\prod_{i=2}^{n}(X + 1 - \frac{\alpha_i}{\alpha}).$$

In the special case of normal extensions we get a relation between the segments of the ramification polygon and the ramifications subgroups of its Galois group.

Remark 2.1.2 (Remark 4.1, [16])
Let L/K be a Galois extension which is generated by an Eisenstein polynomial f with Galois group G. The segments of the ramification polygon $\mathcal{RP}(f)$ of f correspond to the ramification subgroups of G

$$G_j := \{\sigma \in G \mid v_L(\sigma(\alpha)-\alpha) \geq j+1\} \text{ with } j \geq 0.$$

Since $v_L(\frac{\alpha_i-\alpha}{\alpha}) = v_L(\alpha_i-\alpha)-1$ *the ramification polygon describes the filtration*

$$G = G_0 \trianglerighteq G_1 \trianglerighteq \ldots \trianglerighteq G_\kappa = \{\mathrm{id}\}$$

of the Galois group G. *This means that a segment of slope* $-m$ *yields a break at* m *in the filtration, that is* $G_m \neq G_{m+1}$.

By the construction of the ramification polynomial, it is clear that its constant term is equal to zero. This implies that the ramification polygon does not intersect y-axis. Especially, its first leftmost point is $(1, J_0)$, where $n+J_0-1$ is the valuation of $\mathrm{disc}(f)$ (see Figure 1 and Lemma 3.7 in [47]). We can deduce the typical shape of the ramification polygon by the following lemma.

Lemma 2.1.3 ([44], Lemma 1)
Let $f(X) := a_nX^n + \cdots + a_1X + a_0 \in \mathcal{O}_K[X]$ *be an Eisenstein polynomial and* $n = e_0p^w$ *such that* $p \nmid e_0$. *Denote by* α *a root of* f *and set* $L := K(\alpha)$. *Then the following hold for the coefficients of the polynomial* $\psi(X) := \sum_{i=0}^{n} \psi_i X^i := f(\alpha X + \alpha) \in L[X]$:

(a) $v_L(\psi_i) \geq n$ *for all* i.
(b) $v_L(\psi_{p^w}) = v_L(\psi_n) = n$.
(c) $v_L(\psi_i) \geq v_L(\psi_{p^s})$ *for* $p^s \leq i < p^{s+1}$ *and* $s < w$.

The coefficients of ramification polynomial ρ_f are of the form $\rho_i = \sum_{\kappa=i}^{n} \binom{\kappa}{i} a_\kappa \alpha^{\kappa-n}$.
We can easily get that

$$v_L(a_\kappa \alpha^{\kappa-n}) \equiv \kappa \mod n, \ \ \forall\ i \leq \kappa \leq n$$

and thus they are all distinct. Since $v_L(\alpha) = 1$ and $v_L(a_i) \in n\mathbb{Z}$, we have that

$$\begin{aligned} v_L(\rho_i) &= \min_{i \leq \kappa \leq n} \left\{ v_L\left(\binom{\kappa}{i}\right) + v_L(a_\kappa) + \kappa v_L(\alpha) - n v_L(\alpha) \right\} \\ &= \min_{i \leq \kappa \leq n} \left\{ v_L\left(\binom{\kappa}{i}\right) + v_L(a_\kappa) + \kappa - n \right\}. \end{aligned}$$

According to Lemma 2.1.3, we obtain the following corollary for the coefficients of the ramification polynomial.

Corollary 2.1.4
Let $f(X) := a_nX^n + \cdots + a_1X + a_0 \in \mathcal{O}_K[X]$ *be an Eisenstein polynomial and* $n = e_0p^w$ *such that* $p \nmid e_0$. *Denote by* α *a root of* f *and set* $L := K(\alpha)$. *Then the following hold for the coefficients of the ramification polynomial* ρ_f

(a) $v_L(\rho_i) \geq 0$ *for all* i.
(b) $v_L(\rho_{p^w}) = v_L(\rho_n) = 0$.
(c) $v_L(\rho_i) \geq v_L(\rho_{p^s})$ *for* $p^s \leq i < p^{s+1}$ *and* $s < w$.

By Corollary 2.1.4, we can achieve the typical shape of the ramification polygon (see Figure 2.1).

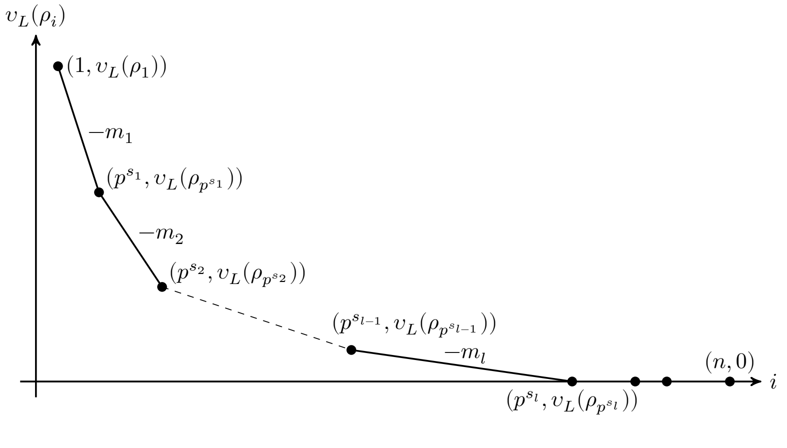

$v_L(\rho_i)$
$(1, v_L(\rho_1))$
$-m_1$
$(p^{s_1}, v_L(\rho_{p^{s_1}}))$
$-m_2$
$(p^{s_2}, v_L(\rho_{p^{s_2}}))$
$(p^{s_{l-1}}, v_L(\rho_{p^{s_{l-1}}}))$
$-m_l$
$(n, 0)$
i
$(p^{s_l}, v_L(\rho_{p^{s_l}}))$

Figure 2.1: Shape of the ramification polygon of an Eisenstein polynomial f of degree n.

Hence, we can compute the ramification polygon of a polynomial without computing its ramification polynomial, which is given in the corollary below.

Corollary 2.1.5
Let $f(X) = \sum_{i=0}^{n} a_i X^i$*, where* $n = e_0 p^r$*, with* $p \nmid e_0$*, be an Eisenstein polynomial. The ramification polygon of* f *is the lower convex hull of the points:*

$$\left\{ \left(p^s, \min_{p^s \le \kappa \le n} \left\{ v_L\left(\binom{\kappa}{p^s} \right) + v_L(a_\kappa) + (\kappa - n) v_L(\alpha) \right\} \right) \ : \ 0 \le s < r \right\} \cup \{(p^r, 0),\ (n, 0)\}$$

in $\mathbb{R}^2$.

Comment 2.1.6
We emphasize that the above corollary is Corollary 4.11 in [15] after making appropriate adjustments.

Example 2.1.7
David Romano in his thesis [41] considers the so-called "Strongly Eisenstein Polynomials". These are polynomials where the coefficient of X *is also a prime element. The definition is given below.*

> **Definition 2.1.8**
> *Let* $f \in \mathcal{O}_K[X]$ *be a monic polynomial of the form* $f(X) = \sum_{i=0}^{n} a_i X^i, \quad a_n = 1$*. The polynomial* f *is called strongly Eisenstein polynomial if* f *is Eisenstein and* $v_K(a_1) = 1$.

Let $f(X) = \sum_{i=0}^{n} a_i X^i \in K[X]$ *be a strongly Eisenstein polynomial of degree* $n = p^r$ *and* $K = K(\alpha)$*, where* α *is a root of* f*. Since* f *is Eisenstein, we have that*

$$v_L(\binom{j}{p^s}) + v_L(a_j) + j - n \ge 1 \ \text{ for } 1 \le j \le n \text{ and } 0 \le s < r.$$

Since $v_L(a_1) = n$*, we can get the equality for* $s = 0$ *and* $j = 1$*. That is*

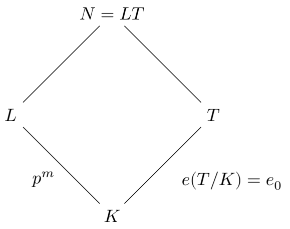
$$v_L(\binom{j}{p^s}) + v_L(a_j) + j_n = v_L(\binom{1}{p^0}) + v_L(a_1) + 1 - n = n + 1 - n = 1.$$

Having that in mind, by Corollary 2.1.5, $\mathcal{RP}(f)$ *consists of exactly one segment connecting the points* $(1, 1)$ *and* $(n, 0)$.

We give the following statements which will be useful for the better understanding of our arguments used in our splitting field approach.

Proposition 2.1.9 (Proposition 4.4, [16])
Let L/K *be a totally ramified extension,* α *be a prime element of* L *and* f *be the minimal polynomial of* α*. Then* $\mathcal{RP}(f)$ *and the segmental inertia degrees of its segments are invariants of* L/K.

Lemma 2.1.10 (Lemma 4.5,[16])
Let L/K *be totally ramified extension of degree* p^m *and let* $-m_1, -m_2, \dots, -m_l$ *be the slopes of* $\mathcal{RP}(L/K)$*. Let* T/K *be the tamely ramified with ramification index* e_0 *and* $N = TL$*. Then the slopes of* $\mathcal{RP}(N/K)$ *are* $-e_0 \cdot m_1, e_0 \cdot m_2, \dots, -e_0 \cdot m_l$.



2.1.1 Computation of Ramification Polygons

At this point we describe in algorithmic form the computation of the ramification polygon of a polynomial. It is essential that in order to compute the ramification polygon we need neither determine the ramification polynomial nor perform calculations over the extension field L. To do this, the Corollary 2.1.5 plays a vital part.

Let us now outline the aforementioned algorithm.

Algorithm 3: Ramification Polygon [RamificationPolygon(f)]

```
Input: A polynomial $f(X) = \sum_{i=0}^{n} a_i X^i \in K[X]$ where $K = \mathbb{F}_q((t))$ and $n = e_0 p^r$.
Output: Compute the ramification polygon of $f$.
1 Determine the points
      $$P_s = \left(p^s, \min_{p^s \le k \le n} \left\{ v_L\left(\binom{k}{p^s}\right) + v_L(a_k) + (k-n)v_L(\alpha) \right\}\right) \text{ for } 0 \le s < r$$
2 Compute the ramification polygon $\mathcal{RP}(f)$ of $f$ as the lower convex hull of the points
      $$\{P_s \mid 0 \le s < r\} \cup \{(p^r, 0),\ (n, 0)\}$$
3 return $\mathcal{RP}(f)$;
```

2.2 Residual Polynomials

Residual (or associated) polynomials were first introduced by Ore [36]. The residual polynomials yield information about the unramified part of the extension generated by a polynomial. They have been used in the factorization of polynomials over local fields [38] and in computing splitting fields for polynomials over local fields [15, 30].

Now we will give the definition and some basic properties of residual polynomials

Definition 2.2.1
Let $f(X) = \sum_{i=0}^{n} c_i X^i$ be a monic polynomial in $\mathcal{O}_K[X]$. Let also

$$S_r : (a_{r-1}, v_K(c_{a_{r-1}})) \leftrightarrow (a_r, v_K(c_{a_r}))$$

be a segment of the $\mathcal{NP}(f)$ whose slope is $\frac{v_K(c_{a_r}) - v_K(c_{a_{r-1}})}{a_r - a_{r-1}} = -h_r/e_r$ with $\gcd(h_r, e_r) = 1$ and set $d_r = a_r - a_{r-1}$ and $b_{r-1} := v_K(c_{a_{r-1}})$. Then the residual polynomial of f that corresponds to S_r is defined to be:

$$\underline{A}_r(y) := \sum_{j=0}^{d_r/e_r} \underline{c_{a_{r-1}+je_r} \pi_K^{jh_r - b_{r-1}}} y^j \in \underline{K}[y].$$

It should be noted that the points $(a_{r-1} + je_r, b_{r-1} - jh_r)$ are the integer points on S_r. In particular, such a point gives a non-zero coefficient of $\underline{A}_r(y)$ if $v_K(a_{r-1} + je_r) = b_{r-1} - jh_r$. Therefore, it leads to the fact that the 0-th and the d-th coefficient of $\underline{A}_r(y)$ are not equal to 0, and so the residual polynomial has degree d and is not divisible by y.

For a detailed description of the derivation of residual polynomials we refer the reader to Section 3, [16] and Section 3.1, Chapter 3, [29].

It is important to note that a factorization of the residual polynomial $\underline{A}_j(y)$ of segment S_j into coprime factors leads to a further factorization of the factor f_j of f that corresponds to S_j.
By construction, it follows immediately the form of the roots of the residual polynomial.

Lemma 2.2.2 (Lemma 3.1, [16])
Let $\beta_1, \dots, \beta_n$ be the roots of $f \in \mathcal{O}_K[X]$. The roots of $\underline{A}(y) \in \underline{K}[y]$ are of the form $\underline{\left(\frac{\beta_i^{e_j}}{\pi_K^{h_j}}\right)}$ for some $1 \le i \le n$ and some $1 \le j \le l$.

Definition 2.2.3
Let $\underline{A}(y) \in \underline{K}[y]$ be the residual polynomial of a segment S of $\mathcal{NP}(f)$ and $\underline{\gamma}$ a root of $\underline{A}(y)$. We call the degree of the splitting field of $\underline{A}(y) \in \underline{K}[y]$ over $\underline{K}$ the segmental inertia degree (or associated inertia) of S.

Furthermore, taking into consideration the representation of the roots of the residual polynomial, it follows that the degrees of the irreducible factors of $\underline{A}_j$ divide the inertia degrees of the extensions generated by the roots of f_j.

Summarizing the above statement and Proposition 1.8.6, we get the following remark.

Remark 2.2.4
The denominators of the slopes (in lowest terms) of the segments of the Newton polygon $\mathcal{NP}(f)$ of f are divisors of the ramification indices of the extensions generated by the irreducible factors of a polynomial. In addition, for each segment of $\mathcal{NP}(f)$ the segmental inertia degree is a divisor of the inertia degree of these extensions.

There is a connection between the ramification polygon of f generating the extension L/K and the ramification polygons of generating polynomials of the subextensions L_{i-1}/L_i. This connection is described in the next theorem.

Theorem 2.2.5 (Theorem 6.2, [16])
For $1 \leq i \leq l+1$ the ramification polygon $\mathcal{RP}(L_{i-1}/L_i)$ consists of exactly one segment, which corresponds to the segment S_i of $\mathcal{RP}(L/K)$ as follows:

(i) The slope of the segment of $\mathcal{RP}(L_{i-1}/L_i)$ is equal to the slope of S_i.

(ii) The segmental inertia degrees of $\mathcal{RP}(L_{i-1}/L_i)$ and S_i are equal.

(iii) For each root $\underline{\delta}$ of the residual polynomial $\underline{A}_i(y)$ of S_i the element $\underline{\delta}^{p^{s_{i-1}}}$ is a root of the associated polynomial of $\mathcal{RP}(L_{i-1}/L_i)$.

2.2.1 Computation of Residual Polynomials

In this section, considering the results of the aforementioned sections: Ramification Polygons (Section 2.1) and Residual Polynomials (Section 2.2), we describe an algorithm for the computation of the residual polynomials. For a deeper discussion of it we refer the reader to Algorithm 4.2 in [15].

Algorithm 4: Residual Polynomials [ResidualPolynomials(f)]

```
Input: A polynomial $f(X) = \sum_{i=0}^{n} a_i X^i \in K[X]$ where $K = \mathbb{F}_q((t))$ and $n = e_0 p^r$.
Output: Compute the residual polynomial for each segment of the Newton polygon of f.
1  Determine the points
       $$P_s = \left(p^s, \min_{p^s \leq k \leq n} \left\{ v_L\left(\binom{k}{p^s}\right) + v_L(a_k) + (k-n)v_L(\alpha) \right\}\right) \text{ for } 0 \leq s < r$$
2  Initialize minList = []. For every point $P_s$ determine $m_s$ to be the index $j$ for which the minimum was
     reached and store it in minList;
3  Compute the ramification polygon $\mathcal{RP}(f)$ of $f$ as the lower convex hull of the points
       $$\{P_s \mid 0 \leq s < r\} \cup \{(p^r, 0),\ (n, 0)\}$$
     and thus specify the segments $S_1, \dots, S_{l+1}$;
4  Define $\delta_0 \leftarrow \underline{(-\pi_K/a_0)} \in \mathbb{F}_q$ and $\delta_1 \leftarrow (\underline{t/\pi_K^{e_K}}) \in \mathbb{F}_q$;
5  Initialize the list polys ← [];
6  for $1 \leq i \leq l$ do
7      for every point $P_s$ lying on the segment $S_i$ do
8          Define $r_2 \leftarrow v_K(a_{m_s})$ and $r_3 \leftarrow v_t\left(\binom{m_s}{p^s}\right)$;
9          Set also $\delta_2 \leftarrow \underline{a_{m_s}/\pi_K^{r_2}} \in \mathbb{F}_q$ and $\delta_3 \leftarrow \underline{\left(\binom{m_s}{p^s}\right)/p^{r_3}} \in \mathbb{F}_q$;
10         Compute the coefficient
               $$\delta_0^{r_3 e_K + r_2} \cdot \delta_1^{r_3} \cdot \delta_2 \cdot \delta_3$$
             of $A_{S_i}(y)$;
11     Construct the polynomial $A_{S_i}(y) \in \mathbb{F}_q[y]$ using its computed coefficients and append it in the list polys;
12 return $\mathcal{RP}(f)$ and the list polys = $[A_{S_1}(y), \dots, A_{S_l}(y)]$ ;
```

Nevertheless, it should be noted that in our case the above coefficient in line 10 can be simplified. In particular, since $K = \mathbb{F}_q((t))$, it yields that $\pi_K = t$ and $e_K = 1$. As a result, $\delta_1 = 1$.

Moreover, we can further analyze the term $r_3 = v_t(\binom{m_s}{p^s})$. Due to the fact that m_s is the index of the summand of the ramification polynomial's coefficient for which the minimum valuation was reached, then $\binom{m_s}{p^s} \not\equiv 0 \,(\mathrm{mod}\ p)$. Hence, $r_3 = v_t(\binom{m}{p^{s_i}}) = 0$.
Considering all the above, the aforementioned coefficient of $A_{S_i}(y)$ is equal to

$$\delta_0^{r_3 e_K + r_2} \cdot \delta_1^{r_3} \cdot \delta_2 \cdot \delta_3 = \delta_0^{r_2} \cdot \delta_2 \cdot \delta_3.$$

2.3 Splitting Fields and Galois Groups in the One Segment Case

2.3.1 Splitting Fields in the One Segment Case

In the event the ramification polygon of an Eisenstein polynomial $f \in \mathcal{O}_K[X]$ consists of one segment, we can quickly determine its splitting field. This case has been analyzed by Greve in Chapter 5 in his PhD-thesis [15] and by Greve and Pauli in [16]. We give the relevant material from [15, 16] without proofs.

According to Lemma 2.1.3, the ramification polygon of an Eisenstein polynomial $f \in \mathcal{O}_K[X]$, of degree $\deg(n) = n$, can consist of exactly one segment either if $p \nmid \deg(f)$ or $\deg(f)$ is a positive power of p, i.e. $n := p^m$, with $m \in \mathbb{N}$. The former case is the tamely ramified case and was studied in the preceding Section 1.9. Thus, we focus on the latter case, i.e $n := p^m$, with $m \in \mathbb{N}$.

The splitting field of f can be described by using the splitting field of its ramification polynomial $\rho_f \in K(\alpha)[X]$, where α is a root of f. To be more precise, the splitting field of ρ_f is a subfield of the splitting field of f. This is given in the next Lemma.

Lemma 2.3.1 (Lemma 7.1, [16])
Assume that the Newton polygon of $\rho_f \in \mathcal{O}_L[X]$ consists of one segment of slope $-h/e$ with $\gcd(h, e) = 1 = ae + bh$ for $a, b \in \mathbb{Z}$ and $\gcd(e, p) = 1$. Assume that its residual polynomial $\underline{A}(y) \in \underline{L}[y]$ is square free and let $\mathfrak{f}$ be its segmental inertia degree. Let I/L be the unramified extension of degree $\operatorname{lcm}(\mathfrak{f}, [L(\zeta_e) : L])$ and let $\epsilon \in \mathcal{O}_I$ with $\underline{A}(-\underline{\epsilon}) = 0$. Then

$$N = I(\sqrt[e]{-(-\epsilon)^b \pi_L})$$

is the splitting field of ρ_f.

Theorem 2.3.2 (Theorem 7.3, [16])
Let $f \in \mathcal{O}_K[X]$ be an Eisenstein polynomial of degree $n = p^m$ and assume that its ramification polygon $\mathcal{RP}(f)$ consists of one segment S of slope $-h/e$ where $\gcd(h, e) = 1 = ae + bh$ for $a, b \in \mathbb{Z}$. Let α be a root of f, $L := K(\alpha)$ and $\underline{A}(y) \in \underline{L}[y]$ be the residual polynomial of $\mathcal{RP}(f)$ with segmental inertia degree $\mathfrak{f}$. Let I/L be the unramified extension of degree $\operatorname{lcm}(\mathfrak{f}, [l(\zeta_e) : L])$ and choose an $\epsilon \in \bar{K}$ such that $\underline{A}(-\underline{\epsilon}) = 0$. Then

$$N := I\left(\sqrt[e]{-\epsilon^b \alpha}\right)$$

is the splitting field of f.

Corollary 2.3.3 (Corollary 5.5, [15])
Let an Eisenstein polynomial $f(X) = \sum_{i=0}^{p^m} a_i X^i \in \mathcal{O}_K[X]$ fulfill the requirements of Theorem 2.3.2. Let also $\underline{A}(y)$ and the numbers b and $\mathfrak{f}$ be as defined in Theorem 2.3.2. Then

$$T = I(\sqrt[e]{(-1)^{p-1} \epsilon^b a_0})$$

is the maximal tamely ramified subfield of the splitting field N, where I/K is the unramified extension of degree $\mathfrak{f}$ and $\epsilon \in \mathcal{O}_V^\times$ with $A(-\underline{\epsilon}) = 0$. Also, we have that

$$N = T(\alpha)$$

where α is a root of f.

The above Corollary can be summarized in the following figure.

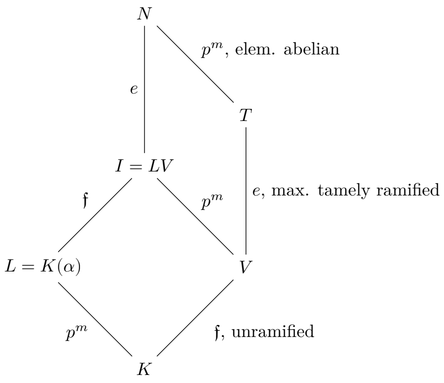


Figure 2.2: **One Segment Case**

It is worth pointing out that Theorem 2.3.2 or Corollary 2.3.3 can be used to compute the splitting field with little computational effort.
Moreover, by Corollary 2.3.3, it holds that the extension N/T is elementary abelian. This is important and it will be highly used in the following sections.

2.3.2 Galois Groups in the One Segment Case

In the case of Eisenstein polynomials whose ramification polygon consists of one segment, we can give an explicit description of its Galois group. If the segment is horizontal then the polynomial generates a tamely ramified extension and its well-known structure of its Galois group is given by Theorem 1.9.3. On the other hand, if the segment is not horizontal, then we can use the results of Section 2.3.1 in order to compute the Galois group. In this section we will study how this can be achieved in more detail.

Let $f \in \mathcal{O}_K[X]$ be an Eisenstein polynomial of degree p^m, α be a root of f and $L := K(\alpha)$ be the subfield of its splitting field generated by α. We denote by N the splitting field of f. In addition, we assume that the ramification polygon $\mathcal{RP}(f)$ of f has one segment S of slope $-h/e$ so that $\gcd(h, e) = 1$ and corresponding residual polynomial $\underline{A}(y) \in \underline{L}[y]$. In this case, we can compute the splitting field N by applying Theorem 2.3.2. Let now T denote the maximal tamely ramified subfield of N/K. Then Theorem 2.3.2 yields that T/K has ramification index e and inertia degree $\mathfrak{f} = \operatorname{lcm}(\mathfrak{f}_1, [L(\zeta_e) : L])$, where $\mathfrak{f}_1$ is the segmental inertia degree of S and ζ_e is an e-th root of unity.

Denote by G the Galois group $G = \operatorname{Gal}(f) = \operatorname{Gal}(N/K)$ and by $\{G_i | i \geq -1\}$ the ramification filtration of G. Then, by Proposition 1.10.5, we get that $G_1 = \operatorname{Gal}(N/T)$. Also, we set $H := \operatorname{Gal}(N/L)$.

By construction, we have that $N = TL$ and $T \cap L = K$. Thus, we have that $G_1 \cap H = \{\mathrm{id}\}$ and $G_1 H = G$. Therefore, $G := G_1 \rtimes H$. Since H is the Galois group of a tamely ramified extension, its structure is well-known by Theorem 1.9.3. It remains to determine the Group G_1 and the action of the elements of H on the elements of G_1.

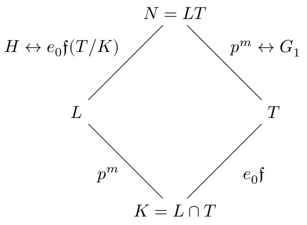

$N = LT$
$H \leftrightarrow e_0\mathfrak{f}(T/K)$
$p^m \leftrightarrow G_1$
L
T
p^m
$e_0\mathfrak{f}$
$K = L \cap T$

Figure 2.3: **Composite of a wildly ramified extension L/K of degree p^m and a tamely ramified extension T/K with ramification index e_0.**

Now we will determine G_1.

Lemma 2.3.4 (Lemma 8.1, [16])
The ramification filtration of $G := \operatorname{Gal}(f)$ is

$$G \geq G_0 \geq G_1 = G_2 = \ldots = G_h > G_{h+1} = \{\mathrm{id}\}.$$

The group $G_1 = \operatorname{Gal}(N/T)$ is isomorphic to the additive group of $\mathbb{F}_{p^m}$.

The next theorem specifies the action of H on G_1 and describes the $\mathrm{Gal}(f)$ as a subgroup of the affine group $\mathrm{AGL}(m, p)$. The unique maximal ideal of $\mathcal{O}_N$ will be denoted by $\wp = \langle \pi_N \rangle$. The group G acts naturally on the additive groups $(\wp^i/\wp^{i+1}, +)$ which are isomorphic to the additive group of $\underline{N}$. For $i \geq 1$ the quotients G_i/G_{i+1} embed into the additive groups $(\wp^i/\wp^{i+1}, +)$, and thus we can define the embedding maps by

$$\begin{aligned} \theta_i : \; G_i/G_{i+1} &\to (\wp^i/\wp^{i+1}, +) \\ \sigma G_{i+1} &\mapsto \left(\frac{\sigma(\pi_N)}{\pi_N} - 1 \right) + \wp^{i+1}. \end{aligned}$$

Theorem 2.3.5 (Theorem 8.2, [16])
Let $f(X) \in \mathcal{O}_K[X]$ be an Eisenstein polynomial of degree p^m, whose ramification polygon consists of one single segment of slope $-\frac{h}{e}$ with $\gcd(h, e) = 1$. Then, $\mathrm{Gal}(f) = G_1 \rtimes H$, where G_1 is the first ramification group and H corresponds to the maximal tamely ramified subfield of the splitting field of f (see Theorem 2.3.2). Moreover, $\mathrm{Gal}(f)$ is isomorphic to the group

$$\tilde{G} = \{t_{A,v} : (\mathbb{F}_p)^m \to (\mathbb{F}_p)^m, X \mapsto AX + v \mid A \in H' \leq \mathrm{GL}(m, p), v \in (\mathbb{F}_p)^m\}$$

of permutations of the vector space $(\mathbb{F}_p)^m$, where H' describes the action of H on $\theta_h(G_h/G_{h+1}) \leq \wp^h/\wp^{h+1}$.

The theorem above is summarized in algorithmic form, which is given below.

Algorithm 5: Galois Group of an Eisenstein polynomial with $\mathcal{RP}$ of one segment [GaloisGroupOneSegment(f)]

```
Input: An irredicible polynomial $f(X) \in \mathcal{O}_K[X]$ of degree $p^m$, whose ramification polygon consists of one
       single segment.
Output: The Galois group of $f$ as a subgroup of $\mathrm{AGL}(m, p)$.

Compute the ramification polygon $\mathcal{RP}(f)$, which consist of one segment $\mathcal{S}$, and the assosiated polynomial
  $\underline{A} \in \underline{L}[X]$, by applying Algorithms 3 and 4, respectively;
Determine the slope $-\frac{h}{e}$ of the segment $\mathcal{S}$;
Define $\mathfrak{f}_1 = \mathrm{lcm}\{\deg g \mid g \mid \underline{A}$ and $g$ is irreducible$\}$ to be the segmental inertia degree of $\mathcal{S}$, and
  $\mathfrak{f} = \mathrm{lcm}(\mathfrak{f}_1, [K(\zeta_e) : K])$;
Compute $a, \tilde{a}, b, \tilde{b} \in \mathbb{N}$ such that $ae - \tilde{a}p^m = 1$ and $bh - \tilde{b}e = 1$;
Generate $\mathbb{F}_{q^\mathfrak{f}}$ and determine a generator $\zeta$ of its multiplicative group $\mathbb{F}_{q^\mathfrak{f}}^\times$;
Compute the roots $\gamma_1, \dots, \gamma_d \in \mathbb{F}_{q^\mathfrak{f}}$ of $\underline{A}$;
Find $r' \in \mathbb{N}$ such that $\zeta^{r'} = -(-u_1^b)$;
Determine $r \in \{0, \dots, e-1\}$ such that $r \equiv r' \mod e$;
Initialize $M \leftarrow \langle 1 \rangle \leqslant \mathbb{F}_{q^\mathfrak{f}}^+$ and $i \leftarrow 1$;
while $\#M = p^m$ do
    Compute the $e$-th roots $\gamma_{i,1}, \gamma_{i,2}, \dots, \gamma_{i,e}$ of $\gamma_i/\zeta^{rh}$;
    $M \leftarrow \langle M, a\gamma_{i,1}, \dots, a\gamma_{i,e} \rangle \leqslant \mathbb{F}_{q^\mathfrak{f}}^+$;
    Increment $i$, that is $i \leftarrow i + 1$;
Choose an $\mathbb{F}_p$-basis $\mathcal{B} = \{b_1, \dots, b_m\}$ of $M$;
Define $l \leftarrow (q^\mathfrak{f} - 1)/e$ and $\kappa \leftarrow r(q-1)/e$;
Determine the automorphism $\sigma$ of $M$ induced by $\zeta^i \mapsto \zeta^{lh+j}$, where $\zeta^j$ is a generator of $M$;
Determine the matrix $S \in \mathrm{GL}(m, p)$ representing $\sigma$ with respect to $\mathcal{B}$;
Determine the automorphism $\tau$ of $M$ induced by $\zeta^i \mapsto \zeta^{h\kappa+qj}$, where $\zeta^j$ is a generator of $M$;
Determine the matrix $T \in \mathrm{GL}(m, p)$ representing $\tau$ with respect to $\mathcal{B}$;
return The Galois group
        $\mathrm{Gal}(f) \cong \{\tau_{A,v} : (\mathbb{F}_p)^m \to (\mathbb{F}_p)^m, X \mapsto AX + v \mid A \in \langle S, T \rangle, v \in (\mathbb{F}_p)^m\}$
  of $f$;
```

It should be noted that as the above algorithm illustrates, the only bits of information required to compute $\mathrm{Gal}(f)$ are: the base field K, the ramification polygon of f and the residual polynomial $\underline{A}$ which corresponds to the segment of the ramification polygon. Every computation step in this algorithm can be performed using data based on those three “objects”.

For a deeper discussion of one segment case we refer the reader to Sections 7 and 8 in [16] or to Sections 3.5 and 3.6 in [29].

2.4 Reduction to p-Extensions

In the general case, when the ramification polygon of an Eisenstein polynomial $f \in K[x]$ consists of more than one segment, the ramification polygon and the residual/associated polynomials do not provide a closed form description of the splitting field of f and its Galois group. However, we can use the ramification polygon of f and its residual polynomials in order to gain information about the structure of its splitting field. In particular, using the concepts and results of the one segment case, we are able to compute a subfield T of the splitting field N of f. Especially, the extension T is the maximal tamely ramified subextension of N/K. That is to say, N is a p-extension of T.

Due to Theorem 2.2.5 we can use the information of the individual segments of $\mathcal{RP}(L/K)$ so as to obtain a subfield T of the splitting field N such that N/T is a p-extension.

It is worth pointing out that we use the results of Section 2.1 to describe the general shape of the ramification polygon. In more detail, the i-th segment S_i is of the form

$$(p^{s_{i-1}}, v_L(\rho_{p^{s_{i-1}}})) \leftrightarrow (p^{s_i}, v_L(\rho_{p^{s_i}})),$$

where the elements ρ_j are the coefficients of the ramification polynomial. The last segment S_{l+1} is horizontal.

The main theorem of this section is the p-Reduction Theorem, whose proof can be found in [16], Theorem 9.1 or [29], Theorem 3.28.

Theorem 2.4.1 (Satz 5.8 [15] / p-Reduction Theorem)
Let $f(X) = X^n + \sum_{i=0}^{n-1} a_i X^i \in \mathcal{O}_K[X]$ be an Eisenstein polynomial of degree $n = e_0 p^m$ with $p \nmid e_0$ and $m > 0$. Assume the ramification polygon $\mathcal{RP}(f)$ of f consists of $l+1$ segments $S_1, S_2, \dots, S_{l+1}$. For $1 \leq i \leq l$ let

- *$m_i = -h_i/e_i$ be the slope of S_i with $\gcd(h_i, e_i) = 1 = d_i e_i + b_i h_i$, for $d_i, b_i \in \mathbb{Z}$,*
- *$A_i(y) \in \mathcal{O}_L[y]$ be the residual/associated polynomial and $\mathfrak{f}_i$ the segmental inertia degree of S_i,*
- *$\gamma_i \in \bar{K}$ such that $\underline{A}(\gamma_i) = 0$, and*
- *$v_i = b_i \cdot e_0 \cdot p^{m-s_{i-1}} + n + 1$.*

Moreover, we denote by U the unramified extension of K of degree

$$\mathfrak{f} = \operatorname{lcm}(\mathfrak{f}_1, \dots, \mathfrak{f}_{l+1}, [K(\zeta_{e_1}) : K], \dots, [K(\zeta_{e_l}) : K])$$

and by N the splitting field of f. Let α be a root of f and

$$K(\alpha) = L_0 \supset L_1 \supset \dots \supset L_l = K$$

be the tower of subfields corresponding to $\mathcal{RP}(f)$ as in Theorem 2.2.5. Then:

a) The field

$$T = U\left(\sqrt[e_1 e_0]{(-1)^{v_1} \gamma_1^{b_1 n} a_0}, \dots, \sqrt[e_l e_0]{(-1)^{v_l} \gamma_l^{b_l n} a_0} \right)$$

is a subfield of N/K, such that N/T is a p-extension.

b) For $1 \leq i \leq l-1$ the extensions TL_{i-1}/TL_i are elementary abelian.

c) The extension T/K is Galois and tamely ramified with ramification index

$$e_0 \cdot \operatorname{lcm}(e_1, \dots, e_l).$$

Furthermore, $[T : K] < n^2$.

We can use the results of the above theorem in order to describe the normal closure N and its relationship to T. This has been summarized in the next Figure.

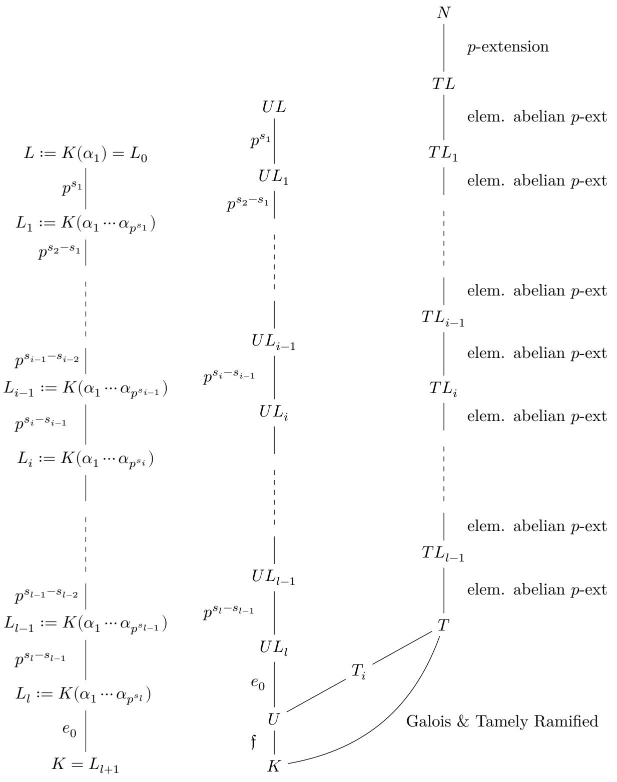


Figure 2.4: p-**Reduction**

Admittedly, Theorem 2.4.1 plays a vital part in the computation of the splitting field of a polynomial. This is due to the fact that the theorem in question contributes to the determination of the field T which is a subfield of the splitting field.

Comment 2.4.2

Having computed the field T, Theorem 2.4.1 implies that we have determined the inertia part $\mathfrak{f}$ such that $\gcd(\mathfrak{f}, p) = 1$. *This means that an additional inertia part $\tilde{\mathfrak{f}}$ such that $p \mid \tilde{\mathfrak{f}}$ can be found after the computation of T.*

For the sake of clarity and completeness of the above Theorem 2.4.1, we express it in an algorithmic form, which is given in the following algorithm.

Algorithm 6: p-Reduction $[p-\text{Reduction}(f))]$

```
Input: An Eisenstein polynomial $f(X) = \sum_{i=0}^{n} a_i X^i \in \mathcal{O}_K[X]$ of degree $n = e_0 p^m$, with $p \nmid e_0$.
Output: An extension field $T$ over $K$ such that the splitting field of $f$ ia a $p$-extension over $T$.

if $p \nmid n$ then
    Create $T = K(\zeta_n, \sqrt[n]{-a_0})$ and return $T$;
Compute the ramification polygon $\mathcal{RP}(f)$ of $f$ and the associated polynomials
    $A_1(Y), \dots, A_l(Y) \in \mathbb{F}_q[Y]$
  of the first $l$ segments by applying the algorithm XFlatRamificationPolygon($f$);
for $1 \leq i \leq l$ do
    Compute the slopes $\frac{h_i}{e_i}$ of the $i-$th segment as well as the gcd representation $d_i e_i + b_i h_i = 1$ and the
      value $v_i$.;
    Determine the segmental inertia degree $\mathfrak{f}_i$ of $S_i$, a root of $\delta_i$ of $A_i(Y) \in \mathbb{F}_{q^{\mathfrak{f}_i}}[Y]$ and $\tilde{\mathfrak{f}}_i = [K(\zeta_{e_i e_0}) : K]$;
Create the unramified extension $U/K$ of degree $\mathfrak{f} = \mathrm{lcm}(\mathfrak{f}_1, \dots, \mathfrak{f}_l, \tilde{\mathfrak{f}}_1, \dots, \tilde{\mathfrak{f}}_l)$;
for $1 \leq i \leq l$ do
    Determine $\epsilon_i \in \mathcal{O}_U^\times$ such that $\underline{\epsilon}_i = \delta_i$;
Create        $T = U\left( \sqrt[e_1 e_0]{(-1)^{v_1} \gamma_1^{b_1 n} a_0}, \dots, \sqrt[e_l e_0]{(-1)^{v_l} \gamma_l^{b_l n} a_0} \right)$;
return $T$;
```

From a computational point of view we would like to explain how we compute the field T, which is a composite of the fields T_i. Note that this composite is only defined if one T_i is normal. By assumption all T_i are normal. We point out that we work over local function fields. For simplicity of the notation we assume that $U = \mathbb{F}_{\tilde{q}}((u_1)) = \mathbb{F}_{\tilde{q}}((t))$.

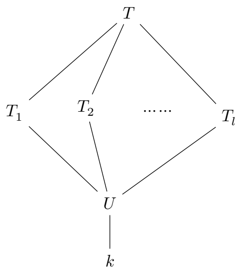


Figure 2.5: **Composition** T_i's

Let $T_i = U(\sqrt[e_i]{\alpha_i})$ for every $1 \leq i \leq l$, then we would like to compute $T := U(\sqrt[e_1]{\alpha_1}, \dots, \sqrt[e_l]{\alpha_l})$. We define $e := \operatorname{lcm}(e_1, \dots, e_l)$ and by assumption we know that $e_i \mid \tilde{q}-1$ and therefore $e \mid \tilde{q}-1$. Suppose by Kummer theory that $T_i = U(z_i)$ such that $y_i = z_i^e$ and $y_i \in U^\times$, and so $T := U(\sqrt[e]{y_1}, \dots, \sqrt[e]{y_l})$.

Using local class field theory, we have that $\operatorname{Gal}(T/U) \leqslant U^\times/(U^\times)^e$. Moreover, by Proposition 1.1.12, we get that

$$U^\times/(U^\times)^e \cong \mathbb{Z}/e\mathbb{Z} \times \mathbb{F}_{\tilde{q}}^\times/(\mathbb{F}_{\tilde{q}}^\times)^e \cong (\mathbb{Z}/e\mathbb{Z})^2 \cong C_e \times C_e.$$

Define the map

$$\varphi : U^\times \to (\mathbb{Z}/e\mathbb{Z})^2, \ y \mapsto (n_0 \mod e, \kappa \mod e),$$

where $y = \sum_{i=n_0}^{\infty} \tilde{y}_i t^i$, with $\tilde{y}_i \in \mathbb{F}_{\tilde{q}}$ and $\tilde{y}_{n_0} = \omega^\kappa$ such that $\tilde{y}_{n_0} \neq 0$. It holds that φ is an epimorphism with $\operatorname{Ker}(\varphi) = (U^\times)^e$.
By Kummer theory we have that

$$\operatorname{Gal}(T/U) = \operatorname{Gal}(T_1 T_2 \cdots T_l/U) \cong \langle y_1, y_2, \dots, y_l \rangle \leqslant U^\times/(U^\times)^e$$

and by applying φ we get

$$G_1 := \langle \varphi(y_1), \dots, \varphi(y_l) \rangle \leqslant C_e \times C_e,$$

and $G_1 \cong \operatorname{Gal}(T/U)$. When G_1 is cyclic of order e then $G_1 = \langle (b, c) \rangle$ and by normalizing it we get $G_1 = \langle (1, c) \rangle$. In the non-cyclic case, by using a normal form algorithm, we can find that $G_1 = \langle (0, a), (1, c) \rangle$, and so $\operatorname{Gal}(T/U) = \langle \varphi^{-1}(0, a), \varphi^{-1}(1, c) \rangle$.

Comment 2.4.3
Specifically, we can have two different shapes of G_1. It can either be of the form $G_1 = \langle (1, c) \rangle$, which means that an additional inertia can not be found, or $G_1 = \langle (0, a), (1, c) \rangle$, which means that the element $(1, c)$ gives the full ramification and the element $(0, a)$ corresponds to the inertia part.

What is left now is to analyze how we compute the defining polynomial of T/U by using G_1. Without loss of generality we compute the composite of two T_i, namely T_1 and T_2.

Remark 2.4.4
Let $G_1 = \langle (0, a), (1, c) \rangle$ as described above. Suppose that $T_i = U(z_i)$ such that $y_i = z_i^e$ for $i = 1\&2$, where $e = \operatorname{lcm}(e_1, e_2)$. Then the composite $T := T_1 T_2 \cong \tilde{U}[X]/\langle g_{T/\tilde{U}} \rangle$, where

$$g_{T/\tilde{U}}(X) = X^e - \omega^c u_1, \ \textit{if } \mathfrak{f}_1 = 1 \quad \textit{or}$$
$$g_{T/\tilde{U}}(X) = X^e - \omega^c u_2, \ \textit{if } \mathfrak{f}_1 > 1,$$

with $[\tilde{U} : U] = \mathfrak{f}_1$, and $\tilde{U} = U$ if $\mathfrak{f}_1 = 1$ and $\tilde{U} \cong \mathbb{F}_{q^{\mathfrak{f}_1}}((u_2))$, otherwise. Also, $\mathbb{F}_{q^{\mathfrak{f}_1}}^\times \cong \langle \omega \rangle$.

Proof. We know that $G_1 \cong \operatorname{Gal}(T/U)$ and $[T : U] = e \cdot \mathfrak{f}_1$, where $\mathfrak{f}_1 \mid e$.

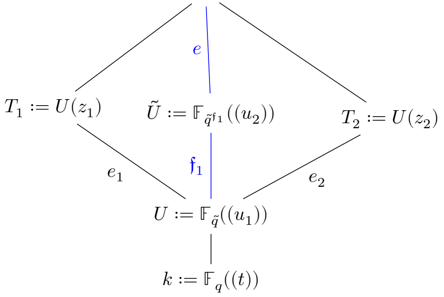

T
e
$T_1 := U(z_1)$
$\tilde{U} := \mathbb{F}_{\tilde{q}^{\mathfrak{f}_1}}((u_2))$
$T_2 := U(z_2)$
e_1
$\mathfrak{f}_1$
e_2
$U := \mathbb{F}_{\tilde{q}}((u_1))$
$k := \mathbb{F}_q((t))$

Figure 2.6: **Composite of** T_1 **and** T_2

If we conclude that $\mathfrak{f}_1 = 1$, then $\tilde{U} = U$, and we are left with the task of determining the generating polynomial of the extension $T/\tilde{U}$. Let denote it by $g_{T/\tilde{U}}$. Additionally, we know that $\deg(g_{T/\tilde{U}}) = e$. On the other hand, if we find out that $\mathfrak{f}_1 > 1$, which means that an additional inertia degree is found, then we change the constant field, as we have constant field extension. Thus, in this case we have that

$$\tilde{U} \cong \mathbb{F}_{q^{\mathfrak{f}_1}}((u_2)) \text{ and } u_2 = u_1,$$

and it remains to determine the defining polynomial of $T/\tilde{U}$. Also, we know that $\deg(g_{T/\tilde{U}}) = e$.

Having computed the group G_1, we have that $G_1 = \langle (0, a), (1, c) \rangle$. We choose the generator of order e, which is the element $(1, c)$. Since we want the generating polynomial to be an Eisenstein, we set

$$g_{T/\tilde{U}}(X) = X^e - \omega^c u_1, \text{ if } \mathfrak{f}_1 = 1, \text{ or}$$

$$g_{T/\tilde{U}}(X) = X^e - \omega^c u_2, \text{ if } \mathfrak{f}_1 > 1.$$

□

We remark that our approach computing the composite of the fields T_i's, described above, has been implemented in a function and used in the splitting field algorithm when needed.

Chapter 3

Splitting Fields over Local Function Fields

3.1 Basic Approach

The splitting field of a polynomial f is the smallest extension field of K in which f has the property to split into linear factors over K. Let $f \in K[X]$, then there exists an extension L/K in which f splits into linear factors, that is

$$f(X) = c(X - \alpha_1)(X - \alpha_2) \cdots (X - \alpha_n),$$

where $\alpha = \alpha_1, \dots, \alpha_n$ are the roots of f and $n := \deg(f)$. The field $L := K(\alpha_1, \dots, \alpha_n)$ which is generated from K by adjoining the roots of f is called the splitting field of f over K. It can be shown that such splitting fields exist and they are unique up to isomorphism. It might happen that the splitting field occurs after fewer than $n - 1$ iterations. In this case, the remaining roots of f are expressible by the adjoined roots.

In this section we would like to compute the splitting field of a polynomial over a local function field of prime characteristic $K := \mathbb{F}_q((t))$, i.e the Laurent series field over a finite field with q elements. We will denote the splitting field of f by $\mathrm{Spl}(f)$.

The basic approach of computing the splitting field of f over local function fields is based on the construction of a sequence of fields

$$K := K_0 \leqslant K_1 \leqslant \dots \leqslant K_{r-1} \leqslant K_r := \mathrm{Spl}(f) \tag{3.1}$$

such that K_i is an extension of K_{i-1} containing a new root of f. We successively adjoin a root of an irreducible factor of the polynomial to the current extension field, and then we refactor the polynomial over the new extension field. We repeat this process until the polynomial $f \in K[X]$ completely factors. The steps for constructing the sequence of fields K_i are the following:

- Firstly, we factorize the polynomial $f \in K_i[X]$ into irreducible factors, let $f := f_1 f_2 \cdots f_{\tilde{r}}$.
- We choose any nonlinear irreducible factor f_i of f.
- We construct the field extension K_{i+1} over K_i as the quotient field $K_{i+1} := K_i[X]/\langle f_i \rangle$, where $\langle f_i \rangle$ is the ideal in $K_i[X]$ generated by f_i.
- We repeat this process for K_{i+1} until the polynomial f factors completely.

It is worth pointing out that we can arbitrarily choose the irreducible factors which are used in the quotient ring construction. Although different choices of irreducible factors may lead to different subfield sequences, the resulting splitting fields will be isomorphic. For the purposes of our implementations, at each step we choose the irreducible factor to be the one with the greatest degree among the possible candidates at that step. In case we have more than one factor of the greatest degree, we arbitrarily choose one of them.

In addition, the degree $[K_{i+1} : K_i]$ of the extension K_{i+1}/K_i is equal to the degree of the irreducible polynomial f_i. Thus, the degree of the extension $\mathrm{Spl}(f)/K$ is $[\mathrm{Spl}(f) : K] = [K_r : K_{r-1}] \cdots [K_1 : K_0] \leq n!$.

Let us now give some technical details for the implementation of this basic approach. We would like to construct extension fields in the following form $\mathbb{F}_q((t))[X]/\langle f \rangle$. To do so, given a polynomial f over series rings, Magma [1] provides the option constructing such an extension by using the factorization code. In order to construct the aforementioned tower of the successively extension fields, we take advantage of the factorization code of the polynomial. In particular, when we factorize the polynomial over the local function field $\mathbb{F}_q((t))$, we use the optional parameter "Extensions". By setting the optional parameter "Extensions" to true, an extension for each factor of the polynomial is returned together with information like the ramification index and the inertia degree of the extension. This way, we construct an extension which is isomorphic to $\mathbb{F}_q((t))[X]/\langle f \rangle$. So, with the intention of extending the current local function field, we use the resulting extension of a non-linear factor of the polynomial as the new extension field in the above tower of fields K_i. We repeat this process until the initial polynomial splits into linear factors.

The above steps of computing the splitting field of a polynomial can be summarized in the following algorithmic form.

Algorithm 7: Splitting Field Algorithm [SplittingField]

```
Input: A polynomial $f$ over $K = \mathbb{F}_q((t))$.
Output: The splitting field of $f$ over $K$, a list with the roots of $f$ and a data which stores
        the minimal polynomial and the corresponding variable in each field extension.
Initialize $i \leftarrow 0$, $L_1 \leftarrow [\,]$, $L_3 \leftarrow [\,]$ and data $\leftarrow [\,]$;
Initialize a list of the roots $R \leftarrow [\,]$ and a list of the non-linear factors NLF $\leftarrow [f]$;
while NLF $\neq \emptyset$ do
    $i \leftarrow i + 1$ and choose $f_1 \leftarrow$ NLF[1] to be the first element of the list of the non-linear
      factors;
    Factorize the polynomial $f_1$ and set the optional parameter "Extensions" to true, then
      $f_1 = f_{1,1} \cdots f_{1,\kappa} \in K[X]$ and $\Gamma_j = [\pi_j, e_j, \mathfrak{f}_j, E_j]$ for every $1 \leq j \leq \kappa$;
    $L_1 \leftarrow [\langle f_{1,j}, \Gamma_j \rangle \,:\, 1 \leq j \leq \kappa \mid \deg(f_{1,j}) > 1]$;
    Choose $f_1 \leftarrow f_{1,1}$ and define $K \leftarrow E_1 \cong K[X]/\langle f_1 \rangle$;
    Roots $\leftarrow [\rho_j \,:\, f_{1,j}(\rho_j) = 0$, with $1 \leq j \leq \kappa \mid \deg(f_{1,j}) = 1]$;
    if $|\tilde{R}| = \deg(f)$ then $R \leftarrow R \cup \tilde{R}$ and break the iteration;
    data $\leftarrow$ data $\cup [\langle f_1, y_i \rangle]$, where $y_i$ is the generating element of $K$;
    Initialize $L_3 \leftarrow [\,]$;
    for $1 \leq j \leq |\mathrm{NLF}|$ do
        Factorize the polynomial $g_j \leftarrow$ NLF[$j$] $\in K[X]$ , that is $g_j = g_{j,1} \cdots g_{j,\kappa} \in K[X]$;
        Compute $L_2 \leftarrow [g_{j,\lambda} \,:\, 1 \leq \lambda \leq \kappa \mid \deg(g_{j,\lambda}) > 1]$ and
          $\tilde{R} \leftarrow [\rho_\lambda \,:\, g_{j,\lambda}(\rho_\lambda) = 0$, with $1 \leq \lambda \leq \kappa \mid \deg(g_{j,\lambda}) = 1]$, and $R \leftarrow R \cup \tilde{R}$;
        Define $L_3 \leftarrow L_3 \cup L_2$;
    Define the list of non-linear factors to be the list $L_3$, that is NLF $\leftarrow L_3$;
return $K, R,$ data;
```

3.2 A Variation of the Basic Approach

The basic approach of computing the splitting field of a polynomial is based on the successive extensions of the local function field. This means that we describe the splitting field by successive extensions and obtain it by adjoining some roots of the given polynomial to the base field. However, the presentation of the splitting field by successive extensions is not practical. From a computational point of view, this representation makes the factorization of a polynomial slow, and consequently, the splitting field algorithm is slow. Moreover, experiments showed that the basic approach might lead to internal errors in Magma of factorization code for some polynomials, as the factorization code over series rings is not the same stable as over global fields.

The bigger the degree $[K_i : K]$ (in the tower (3.1)) is, the more expensive becomes the factorization of the corresponding polynomials. This is true even if we take into account that we factor polynomials of smaller degree by using the already obtained factorizations of the given polynomial f over K_{i-1}.

In other words, the factorization of a polynomial over an extension of the local function field is more expensive than the factorization over a local function field of the form $\mathbb{F}_q((t))$. Thus, we should simplify the description of the splitting field in order to speed up the algorithm. To do so, we had the idea to replace the structure of the complicated successive extension field with a local function field $\mathbb{F}_{\tilde{q}}((u))$ in every step. This will result in achieving quicker factorizations and consequently quicker results. Luckily, this idea is working for the case of the local function fields, and is given below.

Definition 3.2.1
A set R is said to be a set of representatives for a valuation field F if $R \subseteq \mathcal{O}_F$, $0 \in R$ and R is mapped bijectively on $\underline{F}$ under the canonical map $\mathcal{O}_F \to \frac{\mathcal{O}_F}{\mathcal{M}_F} = \underline{F}$.

Definition 3.2.2
*A field $M \subset \mathcal{O}_K$ that is mapped isomorphically onto the residue class field $\underline{M} = \underline{K}$ is called a **coefficient field** in $\mathcal{O}_K$.*

Theorem 3.2.3 (Section (1.1), Chapter IV, [9])
Let L be a local function field with finite residue field $\underline{L} = \mathbb{F}_q$ and $q = p^{\mathfrak{f}}$. Then L is isomorphic to the field of formal power series $\mathbb{F}_q((\pi_L))$, where π_L is a prime element of L.

Proof. Since the residue field $\underline{L}$ is perfect, then there exists a unique coefficient field and it coincides with the set of the multiplicative representatives of $\underline{L}$ in $\mathcal{O}_L$, which is $\mathbb{F}_q$, (see Proposition 5.4, Chapter II in [9]). By Section (5.1), Chapter II in [9], it implies immediately that $L \cong \mathbb{F}_q((\pi_L))$, where π_L is a prime element of L. □

It is worth pointing out that this isomorphism depends on the choice of a coefficient field and the choice of a prime element of L.

Remark 3.2.4
Every totally ramified extension of $\mathbb{F}_p((t))$ is isomorphic to $\mathbb{F}_p((u))$ again.

Now we present our approach of computing the splitting field of a polynomial over a local function field which is based on the idea that every local function field with finite residue field $\mathbb{F}_q$ is isomorphic to the field of formal Laurent series $\mathbb{F}_q((u))$, where u is its prime element.

The basic approach is based on the construction of a tower of fields

$$K = K_0 < K_1 < \dots < K_{r-1} < K_r := \mathrm{Spl}(f)$$

such that $K_i = K_{i-1}(\alpha_i)$, where α_i is a new root of f. Every K_i is a local function field with finite residue field. Thus, according to Theorem 3.2.3, every field K_i is isomorphic to a formal Laurent series field $L_i := \mathbb{F}_{q_i}((u_i))$, where u_i is a prime element of L_i, that is

$$K_i = K_{i-1}[X]/\langle f_i \rangle \cong \mathbb{F}_{q_i}((u_i) = L_i.$$

Therefore, we construct a new tower of fields

$$K := \mathbb{F}_q((t)) = L_0 < L_1 < \dots < L_{\ell-1} < L_\ell := N = \mathrm{Spl}(f),$$

where $L_i := \mathbb{F}_{q_i}((u_i))$ and u_j is embedded in L_{j+1} for every $0 \leq j \leq \ell - 1$ with $u_0 := t$.

$$\begin{tikzcd}
K_r := K(\alpha_1, \alpha_2, \dots, \alpha_r) \arrow[d, dash, dashed] & L_\ell := \mathbb{F}_{q_\ell}((u_\ell)) := N \arrow[d, dash, dashed] \\
K_2 := K(\alpha_1, \alpha_2) \arrow[d, dash] & L_2 := \mathbb{F}_{q_2}((u_2)) \arrow[d, dash] \\
K_1 := K(\alpha_1) \arrow[d, dash] & L_1 := \mathbb{F}_{q_1}((u_1)) \arrow[d, dash] \\
K := \mathbb{F}_q((t)) & K := \mathbb{F}_q((t))
\end{tikzcd}$$

Figure 3.1: **Tower of fields.**

Thus, in every intermediate step we work with the field $\mathbb{F}_{\tilde{q}}((u))$ instead of working with an extension of $\mathbb{F}_q((t))$ by adjoining a root α of the initial polynomial.

$$\begin{tikzcd}
\mathbb{F}_q((t))(\alpha) \arrow[r, phantom, "\cong"] \arrow[d, dash] & \mathbb{F}_{\tilde{q}}((u)) \arrow[d, dash] \\
\mathbb{F}_q((t)) \arrow[r, phantom, "="] & \mathbb{F}_q((t))
\end{tikzcd}$$

In this approach we should express t and α as power series in $\mathbb{F}_{\tilde{q}}((u))$. In other words, we represent t and α as expressions of u.

Let us assume for simplicity that $f = f_1$, and get $K_1 = K[X]/\langle f\rangle$. As explained above, the factorization algorithm produces a uniformizing element π_{K_1} of K_1 and it also computes the degree $\mathfrak{f}$ of the constant field extension. By defining $\tilde{q} := q^{\mathfrak{f}}$ and denote by g_1 the minimal polynomial of π_{K_1} over $\mathbb{F}_{\tilde{q}}((t))$ we get

$$K_1 = K[X]/\langle f\rangle = \mathbb{F}_q((t))(\alpha) \cong \mathbb{F}_{\tilde{q}}((t))[X]/\langle g_1\rangle = \mathbb{F}_{\tilde{q}}((t))(\pi_{K_1}) \cong \mathbb{F}_{\tilde{q}}((u)).$$

Denote by e_1 the ramification index of K_1/K. For the isomorphism

$$\Phi : \mathbb{F}_{\tilde{q}}((t))(\pi_{K_1}) \to \mathbb{F}_{\tilde{q}}((u)),\ \Phi|_{\mathbb{F}_{\tilde{q}}} = \mathrm{id},\ \pi_{K_1} \mapsto u,\ t \mapsto \sum_{j=e_1}^{\infty} b_j u^j,\ \text{where } b_j \in \mathbb{F}_{\tilde{q}},$$

we have to compute the b_j up to some precision which can be done by linear algebra. The only point remaining concerns the value of $\Phi(\alpha)$. Since we use the factorization code to construct the extension field K_1, we have access to the relation between π_{K_1} and the root α. Especially, $\pi_{K_1} = A(\alpha)$ such that $A \in \mathbb{F}_q((t))[X]$, and it yields that $\alpha = B(\pi_{K_1})$ with $B \in \mathbb{F}_q((t))[X]$. The determination of B can be done by linear algebra, which we will explain in detail later in this thesis. Therefore, $\Phi(\alpha)$ is completely defined.

By extending Φ to the polynomial ring via $X \mapsto X$ we can represent all of our factors in the new presentation, and the next factorization steps can be done over this field. Since the right hand field is again a Laurent series field over a finite field, we can use the factoring algorithm over this field. In some sense, we are still factoring over the base field, and therefore the coefficient field of the polynomial ring does not become more complicated during this inductive procedure. Nevertheless, it is clear that some needed t-precision B_1 will be replaced by something like a u-precision of order $e_1 \cdot B_1$. We will study it in the section below.

Let K_1/K be a finite extension and $g \in K_1[X]$ be a polynomial we would like to factor. Then the degree of the polynomial in the new presentation does not change. A factorization using the method of Trager in [50] would lead to factor a polynomial of degree $[K_1 : K] \cdot \deg(g)$ over K. This is the main reason why our new approach is much more efficient than the basic splitting field approach.

The key idea of splitting field computation is to use the new representation in every step of the tower of fields. This yields to compute a uniformizing element of every computed intermediate extension.

Denote by e_i the ramification index of K_i/K_{i-1} and by π_{K_i} the uniformizing element of $\mathcal{O}_{K_i}$. By applying our approach in every step, we inductively get

$$K_i = K_{i-1}[X]/\langle f_i\rangle = \mathbb{F}_{q_{i-1}}((u_{i-1}))(\alpha_i) \cong \mathbb{F}_{q_i}((u_{i-1}))[X]/\langle g_i\rangle = \mathbb{F}_{q_i}((u_{i-1}))(\pi_{K_i}) \cong \mathbb{F}_{q_i}((u_i)),$$

where $q_i = q_{i-1}^{\mathfrak{f}_i}$ with $\mathfrak{f}_i = \mathfrak{f}(K_i/K_{i-1})$ and g_i is the minimal polynomial of π_{K_i} over $\mathbb{F}_{q_i}((u_{i-1}))$. Then we define the isomorphism

$$\Phi_i : \mathbb{F}_{q_i}((u_{i-1}))(\pi_{K_i}) \to \mathbb{F}_{q_i}((u_i)),\ \Phi_i|_{\mathbb{F}_{q_i}} = \mathrm{id},\ \pi_{K_i} \mapsto u_i,\ u_{i-1} \mapsto \sum_{j=e_i}^{\infty} b_j^{(i)} u_i^j, \text{ where } b_j^{(i)} \in \mathbb{F}_{q_i},$$

for every $1 \le i \le \ell$. Also, we have to compute the $b_j^{(i)}$ up to some u_i-precision B_i, which can be done by linear algebra. The only point remaining concerns the behavior of $\Phi_i(\alpha_i)$. Since we use the factorization code to construct the extension field K_i, we have access to the relation between π_{K_i} and the root α_i. Especially, $\pi_{K_i} = A(\alpha_i)$ such that $A \in \mathbb{F}_{q_{i-1}}((u_{i-1}))[X]$, and so $\alpha = B(\pi_{K_i})$ with $B \in \mathbb{F}_{q_{i-1}}((u_{i-1}))[X]$. The computation of B can be done by linear algebra, which we will explain in detail later in this thesis. Therefore, $\Phi_i(\alpha_i)$ is completely defined.

Using $N = K_\ell$, we get that u_ℓ is also a primitive element of N over $\mathbb{F}_{q_\ell}((t)))$, where $\mathbb{F}_{q_\ell}$ is the maximal constant field extension of N/K. Having computed the splitting field N in this presentation, we also have access to all the roots of our given polynomial f, which means that a root of f is of the form

$$\sum_{i=e(N/K)}^{\tilde{B}} a_i u_\ell^i, \text{ with } a_i \in \mathbb{F}_{q_\ell},$$

where $\tilde{B}$ denotes the resulting u_ℓ-precision of N.

Before analyzing the precision, we will describe how we achieve the above described presentation of the uniformizing elements, i.e. $\Phi_i(u_{i-1})$.

Proposition 3.2.5
If K_i/K_{i-1} is an unramified extension of degree $\mathfrak{f}_i$, i.e. a constant field extension in our case, then

$$q_i = q_{i-1}^{\mathfrak{f}_i} \text{ and } \Phi_i(u_{i-1}) = u_i.$$

If K_i/K_{i-1} is a totally ramified extension, then

$$q_i = q_{i-1} \text{ and } \Phi_i(u_{i-1}) = \hat{\rho},$$

where $\hat{\rho}$ is a root of a polynomial $\hat{g}_i \in \mathbb{F}_{q_i}((u_i))[X]$.

Proof. The case of unramified extensions is trivial.
Let K_i/K_{i-1} be a totally ramified extension. Then $\mathfrak{f}_i = \mathfrak{f}(K_i/K_{i-1}) = 1$, and so $q_i = q_{i-1}$. By using the factorization code, we have access to the following extension

$$K_i = K_{i-1}[X]/\langle f_i\rangle \cong K_{i-1}[X]/\langle g_i\rangle = E_i,$$

where $g_i \in \mathbb{F}_{q_{i-1}}((u_{i-1}))[X]$ is an Eisenstein polynomial and u_i is a root of g_i. Let $g_i(X) = \sum_{j=0}^{d_i} b_j X^j \in \mathbb{F}_{q_{i-1}}((u_{i-1}))[X]$, with $b_j = b_j(u_{i-1}) \in \mathbb{F}_{q_{i-1}}((u_{i-1}))$. Consider $g_i(X) = g_i(u_{i-1}, X)$ and by substituting $X = u_i$, we get

$$g_i(u_i) = g_i(u_{i-1}, u_i) = 0 \Leftrightarrow \sum_{j=0}^{d_i} b_j(u_{i-1}) u_i^j = 0 \Leftrightarrow \sum_{j=0}^{\tilde{d}_i} \tilde{b}_j(u_i) u_{i-1}^j = 0$$

which yields that u_{i-1} is a root of the polynomial

$$\hat{g}_i(X) = \hat{g}_i(u_i, X) = \sum_{j=0}^{d_i} b_j(X) u_i^j = \sum_{j=0}^{\tilde{d}_i} \tilde{b}_j(u_i) X^j \in \mathbb{F}_{q_i}((u_i))[X].$$

Therefore, by setting $\Phi_i(u_{i-1}) = \hat{\rho}$, where $\hat{\rho}$ is a root of $\hat{g}_i$, completes the proof. □

Comment 3.2.6
We emphasize that the polynomial $g_i \in L_{i-1}[X]$, with $L_i = \mathbb{F}_{q_{i-1}}((u_{i-1}))$, as defined above, is the irreducible polynomial which generates the extension L_i/L_{i-1}.

It remains to explain how we find the aforementioned root $\hat{\rho}$ of $\hat{g}_i$.

Remark 3.2.7
In order to determine the aforementioned root $\hat{\rho}$ of $\hat{g}_i \in K_i[X]$, where $K_i = \mathbb{F}_{q_i}((u_i))$, we use Hensel's Lemma (Theorem 1.5.1). By applying Hensel's Lemma to $\hat{g}$ with the starting solution to be $\alpha_0 = 0$, (see notation in Hensel's Lemma), we get a root $\hat{\rho} \in \mathcal{O}_{K_i}$ of $\hat{g}$. It is a simple matter to check that we can choose the starting solution to be $\alpha_0 = 0$. In particular, by Hensel's Lemma, we need $\alpha_0 \in \mathcal{O}_{K_i}$ such that $\hat{g}_i(\alpha_0) \equiv 0 \mod u_i$ and $\hat{g}_i'(\alpha_0) \not\equiv 0 \mod u_i$. Then,

$$\hat{g}(\alpha_0) = \sum_{j=0}^{\tilde{d}_i} \tilde{b}_j(u_i)\alpha_0^j = \sum_{j=0}^{d_i} b_j(\alpha_0)u_i^j \equiv b_0(\alpha_0) \mod u_i.$$

Also,

$$\hat{g}_i'(X) = \sum_{j=0}^{d_i} b_j'(X)u_i^j \equiv b_0'(t) \mod u_i, \quad \textit{and so} \quad \hat{g}_i'(\alpha_0) \equiv b_0'(\alpha_0) \mod u_i.$$

Since $\hat{g}_i$ is Eisenstein, then we know that $v_{u_{i-1}}(b_0(u_{i-1})) = 1$, and then for $\alpha_0 = 0$ they hold that

$$\hat{g}(\alpha_0) \equiv b_0(\alpha_0) = b_0(0) = 0 \mod u_i \quad \textit{and} \quad \hat{g}_i'(\alpha_0) \equiv b_0'(\alpha_0) = b_0'(0) \not\equiv 0 \mod u_i,$$

as required.

Hensel's Lemma is a useful tool since it gives us an approximation of a root of the polynomial. One disadvantage of Hensel's Lemma is that it is time-consuming, since it has a lot of expensive computations because of the successively linear liftings of the approximation of the root. In order to speed up the procedure finding an approximation of a root of the polynomial, we can use a quadratic Hensel lifting version. For more details we refer the reader to [21, 22]. It has been observed that the second approach is faster than the first one.

According to quadratic Hensel lifting, we should find $\beta_0 \in \mathcal{O}_K$ such that $b_0(\beta_0) \equiv 0 \mod u_i$. If the polynomial f is Eisenstein, then we can choose the starting solution to be $\beta_0 = 0$.

The quadratic Hensel lifting has already been implemented by Magma and it is called "HenselLift(f, r, prec)". This function lifts the series $r = \beta_0$ as a root of f to precision prec.

As we have described above, we apply the HenselLift algorithm to the polynomial $\hat{g}_i$, which is a polynomial with respect to u_{i-1} arising from the polynomial $g_i(u_{i-1}, u_i)$. So we should swap the variables in the polynomial g_i for achieving $\hat{g}_i$.

Overall, the steps of our approach can be summarized in the next remark.

Remark 3.2.8
Let us denote by $O_i = \mathrm{prec}(u_i)$ the u_i-precision of the field L_i, and by O_0 the t-precision of $K = \mathbb{F}_q((t))$. For every $1 \leq i \leq \ell$ the field L_i can be determined as follows:
If L_i/L_{i-1} is an unramified extension of degree $\mathfrak{f}_i$, i.e. a constant field extension in our case, then

$$L_i := \mathbb{F}_{q_i}((u_i)), \quad \textit{with} \quad q_i = q_{i-1}^{\mathfrak{f}_i}, O_i = O_{i-1} \quad \textit{and} \quad \Phi_i(u_{i-1}) = u_i,$$

where u_i is a uniformizing element of L_i such that $v_{L_i}(u_i) = 1$.
If L_i/L_{i-1} is a totally ramified extension, then

$$L_i := \mathbb{F}_{q_i}((u_i)), \quad \textit{with} \quad q_i = q_{i-1}, \; O_i = O_{i-1} \cdot e(L_i/L_{i-1}) \quad \textit{and} \quad \Phi_i(u_{i-1}) = \hat{\rho},$$

where u_i is a uniformizing element of L_i such that $v_{L_i}(u_i) = 1$ and $\hat{\rho}$ is a root of a polynomial $\hat{g}_i \in \mathbb{F}_{q_i}((u_i))[X]$, as defined in Proposition 3.2.5.

Although we can arbitrary choose the irreducible factors which are used for the tower field construction, we have fixed the choice of the irreducible factor at each step. Precisely, at each step i we choose an irreducible factor with the greatest degree among the possible candidates at that step. In case we have more than one factor of the greatest degree, we arbitrarily choose one of them. Once we choose an irreducible factor, we continue to construct the next fields L_i of the tower by using its possible factors until this factor splits into linear factors. Then we proceed to the next already computed irreducible factor of the original polynomial with the greatest degree and repeat the same strategy. We repeat this process until the original polynomial factors completely.

Furthermore, it is worth pointing out that during our process, once we find out that a non-trivial inertia degree occurs, we construct the unramified extension and then we proceed by choosing an irreducible factor according to the above described strategy.

The only point remaining concerns the determination of the precision, which we will study in the next section.

3.2.1 Precision

A technical issue that we should figure out is the choice of the precision in each intermediate step of the tower of the successive extensions for the needs of splitting field computation. Precision plays a vital part in our splitting field algorithm, as the latter is based on the factorization of polynomials. Especially, the given precision in every step plays a crucial role in the successful factorization of a polynomial. Therefore, it is essential to give as much precision as needed so as to achieve the factorization. On the other hand, the strategy to give large precision is not practical, since from a computational point of view this way causes a time-consuming factorization of a polynomial, and so in general an expensive algorithm. Thus, we have to set the best possible minimal bound for the required precision in every case such that it leads to correct results.

Assume that $f \in \mathbb{F}_q((t))[X]$ is given with precision equal to prec. Then, according to the description of our splitting field algorithm, we construct a tower of fields, (see Figure 3.1),

$$K := \mathbb{F}_q((t)) = L_0 < L_1 < ... < L_{\ell-1} < L_\ell := \mathrm{Spl}(f) = L, \tag{3.2}$$

where $L_i := \mathbb{F}_{q_i}((u_i))$ and $\Phi_1(t) \in L_1$, $\Phi_{j+1}(u_j) \in L_{j+1}$, with $1 \leq j \leq \ell - 1$. In addition, for every intermediate step $L_{i-1} < L_i$ there exists an irreducible polynomial $g_i \in L_{i-1}[X]$ which generates the extension L_i/L_{i-1}.

For a start, let us denote the u_i-precision at the step i by O_i. Then, according to the induced bound of the precision, as we analyzed above, we set the u_i-precision at the step i to be equal to the product of the u_{i-1}-precision at the step $i-1$ with the ramification index of the extension L_i/L_{i-1}. Note that the ramification index of the extension L_i/L_{i-1} is the degree of the defining polynomial of the extension L_i/L_{i-1} which is $\deg(g_i)$. Particularly, $\deg g_i = e(L_i|L_{i-1}) = e_i$. That is to say, for every $1 \leq i \leq \ell$ the u_i-precision follows the following rule

$$O_i = O_{i-1} \deg(g_i),$$

where $O_0 = $ prec is the precision of K. In this situation, obviously, that choice of the precision at each step results in a successful factorization, and so in a feasible algorithm.

Nevertheless, it should be noted that the above strategy for the precision can end up in extremely large values of the precision. As a result, the factorization would be expensive, and thus, the splitting field algorithm would be slow. A satisfactory solution for this problem would be the determination of a more efficient value of the precision in each step provided that it will bring about successful factorization.

Now we describe a way for the choice of the precision using information that we can obtain by the ramification polygon of the polynomial in question. As we have already described, the ramification polygon is a useful tool for the computation of the splitting field.

Especially, we determine the required precision in every step by taking advantage of the valuation of the differences of the roots of f, that is the valuation of the elements $\alpha_1 - \alpha_i$ with $2 \leq i \leq n$. With the purpose of computing the valuation of such elements we use the ramification polygon of the polynomial f. Particularly, by Definition 2.1.1 we know that the roots of the ramification polynomial ρ_f of f are of the form $\frac{\alpha_i - \alpha}{\alpha}$, with $1 \leq i \leq n$, where $\alpha = \alpha_1, \cdots, \alpha_n$ are the roots of f. We remind that $v_L = v_{u_\ell}$ stands for the valuation of L that is normalized such that $v_L(u_\ell) = 1$, where u_ℓ is the uniformizing element in $\mathcal{O}_L$. Hence,

$$v_L(\frac{\alpha_i - \alpha}{\alpha}) = v_L(\alpha_i - \alpha) - v_L(\alpha) \Rightarrow v_L(\alpha_i - \alpha) = v_L(\frac{\alpha_i - \alpha}{\alpha}) + v_L(\alpha).$$

According to Proposition 1.8.3, we can compute the valuation of the roots of the ramification polynomial by using the slopes of its ramification polygon (i.e Newton polygon of ramification polynomial). Let us denote $\tilde{L} := K(\alpha)$ and its valuation by $v_{\tilde{L}} = v_\alpha$ such that $v_{\tilde{L}}(\alpha) = v_\alpha(\alpha) = 1$. Suppose that $-m_1 < \cdots < -m_l$ are the slopes of the ramification polygon $\mathcal{RP}(f)$ with respect to $\tilde{L}$-valuation, where $\mathcal{RP}(f) = \{(1, v_{\tilde{L}}(\rho_1)), (p^{s_1}, v_{\tilde{L}}(\rho_{p^{s_1}})), (p^{s_2}, v_{\tilde{L}}(\rho_{p^{s_2}})), \dots, (p^{s_l}, v_{\tilde{L}}(\rho_{p^{s_l}})), (n, 0)\}$.

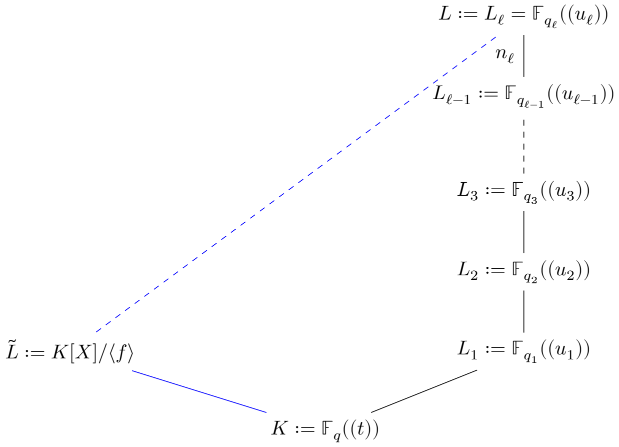

$L := L_\ell = \mathbb{F}_{q_\ell}((u_\ell))$
n_ℓ
$L_{\ell-1} := \mathbb{F}_{q_{\ell-1}}((u_{\ell-1}))$
$L_3 := \mathbb{F}_{q_3}((u_3))$
$L_2 := \mathbb{F}_{q_2}((u_2))$
$\tilde{L} := K[X]/\langle f \rangle$
$L_1 := \mathbb{F}_{q_1}((u_1))$
$K := \mathbb{F}_q((t))$

Figure 3.2: **Splitting Field Tower**

We know that the roots of the ramification polynomial have $\tilde{L}$-valuation equal to δ, where $\delta \in \{m_1, m_2, \dots, m_l\}$, i.e. $v_{\tilde{L}}(\frac{\alpha_i - \alpha}{\alpha}) = \delta$ and $\nu_{\tilde{L}_{\max}}$ denotes the maximum of them. So then $\nu_{\tilde{L}_{\max}} = \max\{\delta_i \ : \ 1 \leq i \leq l\} + 1$. Let us denote by M_i the minimum precision is needed to ensure that $\alpha_i - \alpha_j \not\equiv 0 \mod u_i^{M_i}$. By the above discussion, we can set M_i as indicated in the remark below.

Remark 3.2.9

Under the conditions stated above, we set $M_1 = \max\{\delta_i \ : \ 1 \leq i \leq l\} + 2$ and $M_i = e(L_i/K)M_1$ for every $i > 1$.

Experiments drive us getting the following heuristic result for setting an optimal value for the precision needed at every step. The next result summarizes our approach to the precision determination and the gradually computation of $v_L(\alpha)$.

Heuristic Assumption 3.2.10

In every step i we increase the u_i-precision in the following way:

- *If $i = 1$, then we set $v_{L_1}(\alpha) := 1$ and $O_1 := e(L_1/K)M_1 + e(L_1/K)v_{\tilde{L}}(\rho_1) + 50$.*
- *If $i > 1$, then we set $v_{L_i}(\alpha) := v_{L_{i-1}}(\alpha)\deg g_i$ and $O_i := O_{i-1} + e(L_i/K)M_1 + 20$.*

For the sake of completeness the results and theory above are summarized in algorithmic form, which is given below.

Algorithm 8: Precision Determination, [PrecisionDetermination]

```
Input: Given $O_{i-1}, e = e(L_i/L_{i-1}), \mathcal{RP}, [m_1, \dots, m_l], O_{\mathcal{RP}}, v_{L_{i-1}}(\alpha), i, f, p$ defined as above.
Output: The precision $O_i$ at step $i$ and $v_{L_i}(\alpha)$.
if $i=1$ then $O_i \leftarrow e(L_1/K)M_1 + e(L_1/K)v_{\tilde{L}}(\rho_1) + 50$ and $v_{L_i}(\alpha) \leftarrow 1$ ;
else $v_{L_i}(\alpha) \leftarrow v_{L_{i-1}}(\alpha)e(L_i/L_{i-1})$ and $O_i \leftarrow O_{i-1} + e(L_i/K)M_1 + 20$;
return $O_i, v_{L_i}(\alpha)$;
```

Now we are able to present the algorithm of the splitting field computation.

Algorithm 9: Splitting Field Algorithm (SplittingFieldLFF)]

```
Input: A polynomial $f \in k[X]$, where $k = \mathbb{F}_q((t))$ of precision $O_0$.
Output: The splitting field of $f$ over $K$, the order of the Galois group, a list with the roots of $f$, a list
        $[\Phi_1(t) \in L_1, \Phi_{j+1}(u_j) \in L_{j+1},$ with $1 \leq j \leq \ell - 1]$ and a dataset with the minimal polynomial in
        each intermediate field extension.
Initialize prec $\leftarrow O_0$, Lin_Fact $\leftarrow [\,]$, NonLin_Fact $\leftarrow [\,]$, LF $\leftarrow [\,]$, data $\leftarrow [\,]$, subst $\leftarrow [\,]$,
  RootsASubst $\leftarrow [\,]$, $i \leftarrow 0$, deg $\leftarrow 1$, $u_0 \leftarrow t$, $\tilde{q}_0 \leftarrow q$, $k_0 \leftarrow k$ and $\hat{f} \leftarrow f$;
Applying RamificationPolygon algorithm, determine
  $\mathcal{RP} \leftarrow \{(1, v_{\tilde{L}}(\rho_1)), (p^{s_1}, v_{\tilde{L}}(\rho_{p^{s_1}})), (p^{s_2}, v_{\tilde{L}}(\rho_{p^{s_2}})), \dots, (p^{s_l}, v_{\tilde{L}}(\rho_{p^{s_l}})), (n, 0)\}$,
  slopes $= [m_1 > m_2 > \cdots > m_l]$ and $D_1 := \gcd(v_{\tilde{L}}(\rho_1), n-1)$;
repeat
    $i \leftarrow i + 1$;
    Factorize the polynomial $\hat{f}$, namely
            $\hat{f} = \hat{f}_1 \cdots \hat{f}_\kappa \in k_{i-1}[X]$ and $\Gamma_j = [\pi_j, e_j, \mathfrak{f}_j]$ for every $1 \leq j \leq \kappa$;
    Create  NonLinFact $\leftarrow [[\hat{f}_j, \Gamma_j] \,:\, 1 \leq j \leq \kappa \mid \deg(\hat{f}_j) > 1]$ and
            LF $\leftarrow [[\hat{f}_j, \mu(\hat{f}_j), i-1] \,:\, 1 \leq j \leq \kappa \mid \deg(\hat{f}_j) = 1]$;
    Update the list of linear factors Lin_Fact $\leftarrow$ Lin_Fact $\cup$ LF;
    if $|$Lin_Fact$| = \deg(f)$ then break the iteration;
    Choose the new $\hat{f}$ being a non-linear factor of $f$, provided that $|$NonLinFact$| > 0$. This means that
      $\hat{f} \leftarrow \hat{f}_1$. Otherwise, choose $\hat{f} \leftarrow$ NonLin_Fact[1][1] and define NonLinFact $\leftarrow$ [NonLin_Fact[1]];
    Take the inertia degree $\mathfrak{f}(k_i/k) := \operatorname{lcm}([\mathfrak{f}_j \,:\, 1 \leq j \leq \kappa])$;
    if $\mathfrak{f}(k_i/k) > 1$ then
        Change the constant field, namely $F_{\tilde{q}_i} \leftarrow \mathbb{F}_{\tilde{q}_{i-1}^{\mathfrak{f}(k_i/k)}}$, $F_{\tilde{q}_i}^\times = \langle w \rangle$ and define $k_i := \mathbb{F}_{\tilde{q}_i}((u_i))$ with
          precision prec. Also define deg $\leftarrow$ deg $\cdot \mathfrak{f}(k_i/k)$;
        Update the coefficients of the polynomial $\hat{f}$ in the new ring $k$ and append the generating element
          of $k_i$ in the list of substitutions, that is subst := subst $\cup \{u_i\}$. Define the polynomial ring $k_i[X]$;
        Update the lists of non-linear NonLin_Fact and linear factors Lin_Fact of $f$ by checking the
          non-linear factors arising from the preceding steps NonLin_Fact and by checking the non-linear
          factors arising from the current step NonLinFact;
        Continue for the next iteration;
    else
        Factorize the polynomial $\hat{f}$ by using the optional parameter "Extensions", then
          $\hat{f} = \hat{g}_1 \cdots \hat{g}_\kappa \in k_{i-1}[X]$ and $\Gamma_j = [\pi_j, e_j, \mathfrak{f}_j, E_j \cong k_{i-1}[X]/\langle \tilde{g}_j \rangle]$ for every $1 \leq j \leq \kappa$;
        Define $f_1 \leftarrow \tilde{g}_1$, $g_i \leftarrow \hat{f}$, and data $\leftarrow$ data $\cup [[g_i,\ f_1]]$;
        Determine $e(k_i/k) \leftarrow \deg(f_1)$ and prec, $v_{L_i}(\alpha)$ by applying Algorithm 8;
        Set $k_i \leftarrow \mathbb{F}_n((u_i))$ with precision prec.
    Take deg $\leftarrow$ deg $\cdot e(k_i/k)$ and define the polynomial ring $k_i[X]$. Afterwards, consider the polynomial $f_1$
      as a polynomial of $u_i$, and consider $f_1 \leftarrow f_1 \mod u_i^{\text{prec}}$;
    Apply the algorithm of Hensel's Lift on $f_1$, namely $\mathfrak{s}_i \leftarrow$ HenselLift$(f_1, 0, \text{prec}) \in k_i$;
    Append the $\mathfrak{s}_i$ in the list of substitutions subst, subst $\leftarrow$ subst $\cup \{\mathfrak{s}_i\}$ and consider
      $\hat{f} = \hat{f}(\mathfrak{s}_i, X) \in k_i[X]$;
    Update the lists of non-linear NonLin_Fact and linear factors Lin_Fact of $f$ by checking the
      non-linear factors arising from the preceding steps NonLin_Fact, and by checking the non-linear
      factors arising from the current step NonLinFact;
    Continue for the next iteration;
until $|$Lin_Fact$| = \deg(f)$;
for $1 \leq j \leq |$Lin_Fact$|$ do
    if $l_j :=$ Lin_Fact[$j$][2] $< |$subst$| := \lambda$ then
        Define $\hat{g} \leftarrow$ Lin_Fact[$j$][1] $:= X - \tilde{\rho}_j$;
        Define $\Phi := \Phi_{l_j+1} \circ \Phi_{l_j+2} \circ \cdots \circ \Phi_\lambda$ and $\rho_j \leftarrow \Phi(\tilde{\rho}_j)$;
        Append in the list RootsASubst the root $\rho_j$ of $\hat{g}$, that is RootsASubst $\leftarrow$ RootsASubst $\cap [\rho_j]$;
    else
        Append in the list RootsASubst the root $\rho_j$ of the original linear factor, that is
          RootsASubst $\leftarrow$ RootsASubst $\cap [\rho_j)]$;
if $|$data$| = 0$ then define data := $[[f,\ f]]$ and subst := $[t]$;
return $k$, deg, RootsASubst, subst,data;
```

In the above algorithm, in order to consider $\hat{f} = \hat{f}(\mathfrak{s}_i, X) \in k_i[X]$, we use the function SubstHL(f, elm), which takes as input a polynomial $f = f(t, X) \in \mathbb{F}_q((t))[X]$ and an element elm $\in \mathbb{F}_{\tilde{q}}((u))$, and returns $f(\text{elm}, X)$. In our case, elm is the representation of t over $\mathbb{F}_{\tilde{q}}((u))$. The list LF contains the 3-tuples $[h,\ \mu(h), i]$, where h is a linear factor, the positive integer $\mu(h)$ is its multiplicity and the positive integer i is the number of the step in which h is achieved. The list Lin_Fact is constructed by the list LF and it consists of the pairs $[h,\ i]$.

Chapter 4

Factorization

In this section we will describe an improvement for the computation of the factorization using the computer algebra system Magma [1]. For a start, let us analyze some of the facts behind the factorization code provided by Magma. Factorization is available for polynomials over series rings defined over finite fields. When we factorize the polynomial over the local function field $\mathbb{F}_q((t))$ we can use the optional parameter "Extensions". By setting that parameter to true, an extension for each factor of the polynomial is returned together with information like the ramification index and the inertia degree of the extension. In order to compute that extension, it is important to compute its defining polynomial. To do so, the function called "Chi" is used. To be more precise, what that function does is nothing else but computing the defining polynomial of the desired extension.

It is important to further examine that function "Chi". In the case of fields with characteristic zero or large enough positive characteristic, that function uses the Newton relations, which provide a relation between the basic symmetric functions and the higher traces (i.e. powers sums of the roots), to compute the desired defining polynomial of the extension. However, in case of fields of small positive characteristic, they employ the function called "oldchi" to determine the required defining polynomial. In more detail, the latter function is based on the kernel version, and as a result it is time consuming for local function fields with large precision.

Therefore, it is natural to try to relate the case of fields with small positive characteristic to that of zero or large enough positive characteristic. In other words, we are interested in using an approach based on Newton Relations for the fields with small positive characteristic as well. Almost all of our interesting examples are wildly ramified and thus the degree of the polynomial is greater than the characteristic p.

4.1 Basic Symmetric Functions and Higher Traces of Roots

The key idea to our approach is to represent a polynomial by using the higher traces (i.e. power sums) of the roots and vice versa. That is to say, given the higher traces of the roots, then reconstruct the polynomial from them. This can be found in [8].

It is worth reminding that the Newton relations provide a straightforward way to make these conversions. In particular, the correspondence between the basic symmetric functions and the higher traces (powers sums) of the roots is given in the following theorem.

Let us first recall the definition of the higher traces (power sums).

Definition 4.1.1
Let $\alpha_1, \dots, \alpha_n$ be the roots of a polynomial in some algebraic closure. Then we will denote by $S_\kappa := \sum_{i=1}^{n} \alpha_i^\kappa$ the κ-th Higher Trace (power sum) of $\alpha_1, \dots \alpha_n$, where $\kappa \in \mathbb{Z}_{\geq 0}$.

Definition 4.1.2
The basic symmetric functions in n variables $\xi_1, \dots, \xi_n$ for $\kappa = 1, \dots, n$ are defined by

$$\sigma_\kappa(\xi_1, \dots, \xi_n) := \sum_{1 \leq j_1 < \cdots < j_\kappa \leq n} \xi_{j_1} \xi_{j_2} \cdots \xi_{j_\kappa}.$$

Comment 4.1.3
Let R be a commutative ring. Any polynomial in n variables $\xi_1, \dots, \xi_n$ over R which remains unchanged for any permutation of the variables can be represented as a polynomial in the basic symmetric functions $\sigma_\kappa(\xi_1, \dots, \xi_n)$ over R.

Thus, now we can give the Newton Relations.

Theorem 4.1.4 (Theorem 3.12, [39])
Let $\alpha_1, \dots, \alpha_n$ be the roots (repeated with multiplicity) of a monic polynomial. Then, between the basic symmetric functions and the higher traces we have the so-called Newton relations:

$$\sum_{i=0}^{\kappa-1}(-1)^i\sigma_i S_{\kappa-i} + (-1)^\kappa \kappa\sigma_\kappa = 0, \quad \textit{with } \sigma_0 := 1,\ 0 \le \kappa \le n, \tag{4.1}$$

$$\sum_{i=0}^{n}(-1)^i\sigma_i S_{\kappa-i} = 0, \quad \textit{with } \sigma_0 := 1,\ \kappa \ge n, \tag{4.2}$$

It is a fact that Newton's Relations allow us to determine the higher traces by using the basic symmetric functions and vice versa. For instance,

$$\begin{array}{lcl} S_1 = \sigma_1 & & \sigma_1 = S_1 \\ S_2 = \sigma_1^2 - 2\sigma_2 & \text{and} & \sigma_2 = \frac{1}{2}(S_1^2 - S_2) \\ S_3 = \sigma_1^3 - 3\sigma_1\sigma_2 + 3\sigma_3 & & \sigma_3 = \frac{1}{6}S_1^3 - \frac{1}{2}S_1S_2 + \frac{1}{3}S_3 \end{array}$$

provided that 2 and 3 are invertible in the working ring. Note that the one direction always works. Only the other direction is a problem in small characteristic.

Theorem 4.1.4 enables us to have access to the power sums of the roots of a polynomial. Conversely, using Theorem 4.1.4 and bearing in mind that a polynomial can be represented as a polynomial in the basic symmetric functions (see Comment 4.1.3), we can construct a polynomial given the power sums of the roots only in characteristic zero or large enough positive characteristic. Thus, the above results give us the translation between a polynomial and the power sums of its roots.

In the next paragraph we will represent another approach for the above translation.

4.2 Fast Conversion between Polynomials and Higher Traces of Roots

In [3, 2] the authors proposed another approach for the conversions between a monic polynomial and the power sums of its roots, which is worth discussing. Let us now describe their approach.

Before analyzing the two directions, we establish some notation. In the rest of the chapter, we will make use of the following notation: Given $f \in k[T]$ of degree $\deg(f) = D$,

- $\mathrm{rev}(f)$ stands for its reversal, that is $\mathrm{rev}(f) = T^{\deg(f)} f(\frac{1}{T})$. Especially, if f is of degree at most n, then we write
$$\mathrm{rev}(n, f) = T^n f(\frac{1}{T})$$
for its n-th reversal.
- we write $\mathrm{Newton}(f)$ for the Newton Series of f which is the power series
$$\mathrm{Newton}(f) = \sum_{\kappa \ge 0} S_\kappa(f) T^\kappa \in k[[T]].$$
- Given a power series $S = \sum_{i \ge 0} s_i T^i \in k[[T]]$ and an integer $m \ge 1$, we write $S \mod T^m$ for the truncated power series $\sum_{i=0}^{m-1} s_i T^i$.
- Given $0 \le l < h$, we use the operations $\lceil \cdot \rceil^h$, $\lfloor \cdot \rfloor_l$ and $[\cdot]_l^h$ on $P = \sum_{i=0}^{n} p_i T^i$ and defined as follows:
$$\lceil P \rceil^h = \sum_{i=0}^{h-1} p_i T^i, \quad \lfloor P \rfloor_l = \sum_{i=0}^{n-l} p_{i+l} T^i, \quad [P]_l^h = \sum_{i=0}^{h-l-1} p_{i+l} T^i.$$

As we have mentioned, the speed-up of factorization can be achieved and this behavior is based on the use of an alternative encoding for univariate polynomials, which is the Newton representation by the power sums of roots. The use of such an encoding for polynomials by using Newton representation is classical and has widely presented in literature (see for instance [8, 45, 51]). Moreover, in [3, 2] the authors proved that the use of Newton representations for polynomials provides the appropriate data structure for the efficient computation of composed and diamond operations.

It is worth bearing in mind that in case of characteristic is equal to zero or larger than D, any polynomial of degree D is uniquely determined by the first D higher traces of its roots. In particular, as we described above, Newton relations provide a straightforward algorithm to perform these conversions. However, its complexity is quadratic in D. On the other hand, faster conversion methods have been developed due to Schönhage and Pan, and thus, we will recall their algorithms.

4.2.1 From Polynomials to Newton Representations

For a start, we present an algorithm for the direct direction, namely, from a polynomial to its Newton representation. Schönhage [46] was the first to propose an efficient algorithm for this case. The translation from a polynomial to the power sums of its roots is quite simple and is based on the next lemma. It is important to point out that this works in arbitrary characteristic.

Lemma 4.2.1 (Lemma 1, [3], [2])
Let $f \in k[X]$ be a monic polynomial of degree $\deg(f) = D$. Then the series $\mathrm{Newton}(f)$ is rational. Moreover, the following formula holds:

$$\mathrm{Newton}(f) = \frac{\mathrm{rev}(D-1, f')}{\mathrm{rev}(D, f)}.$$

In the following result, the complexity of the algorithm for the direct direction is given.

Proposition 4.2.2 (Proposition 1, [2])
If $f \in k[X]$ is a polynomial of degree D in $k[X]$ and $N \geq D$, then the first N power sums of the roots can be computed within $O(\frac{N}{D}M(D))$ operations in base field k, where $M(D)$ stands for the number of operations in k required to perform the product of two polynomials of degree at most D.

The corresponding algorithm is summarized in the following result.

Algorithm 10: Computing the Newton Series of a polynomial

```
Input: A polynomial $f \in k[X]$ of degree $D$
Output: $\mathrm{Newton}(f)$ at precision $N \geq D$.
$A \leftarrow \mathrm{rev}(D-1, f')$;
$B \leftarrow \mathrm{rev}(D, f)$;
$B_0 \leftarrow \lceil \frac{1}{B} \rceil^D$;
$C_0 \leftarrow \lceil AB_0 \rceil^D$;
$l \leftarrow \lfloor \frac{N}{D} \rfloor$;
for $j; = 0$ to $l$ do
    $C_{j+1} = -\lceil \lfloor BC_j \rfloor_D B_0 \rceil^D$
return $\sum_{i=0}^{l} C_i T^{Di} + O(T^N)$;
```

Note that C_i in the above algorithm is a polynomial in k of degree less than D.

4.2.2 From Newton Representations to Polynomial Representations

Let us proceed to the converse direction, which is more difficult to handle. While in characteristic zero case the Newton relations give a one-to-one correspondence between power sums and basic symmetric functions, this is not true in the positive characteristic case. Precisely, in the latter case distinct monic polynomials of the same degree may have equal power sums of roots. Some typical examples are X^2 and X^2+1 over $\mathbb{F}_2$ or X^2+t and X^2+1 over $\mathbb{F}_2((t))$ or X^2+tX+t and

X^2+tX+t^2 over $\mathbb{F}_2((t))$. Consequently, our approach will rely on the characteristic of the base field.

In particular, we distinguish two cases, which were studied by Schönhage in [46] and Pan in [37], and are summarized in the following proposition.

Proposition 4.2.3 (Proposition 1, [3])
Let $f \in k[X]$ be a polynomial of degree D in $k[X]$. Then:

1. *If the characteristic of k is equal to zero or greater that D, then the polynomial f can be computed from the first D powers sums of its roots within $O(M(D))$ operations in the base field k.*
2. *If k has positive characteristic p and if all the roots of f have multiplicities less than p, then the polynomial f can be computed from the first $2D$ power sums of its roots within $O(M(D)\log(D))$ operations in the base field k.*

In the first part of the above Proposition 4.2.3, we treat the case of characteristic zero or large enough, for which the solution is based on the existence of the exponential of a power series. For a thorough treatment in that case we refer the reader to Subsection 2.1 in [3] and to Subsection 2.2.1 in [2]. The second one refers to the arbitrary positive characteristic case. In that case and under some mild assumptions, the solution relies on the so-called Berlekamp-Massey algorithm (see [11] for more details). We will thoroughly look into it below.

4.3 The Arbitrary Positive Characteristic Case

In this section we will focus on analyzing the second part of the Proposition 4.2.3, which will be used for describing an algorithm. The conversion from power sums to coefficients of a polynomial in small characteristic is a more subtle problem. We begin with introducing some terminology.

Definition 4.3.1
A sequence $\mathcal{S} = (s_i)_{i=0}^{\infty}$ of an arbitrary field F is called a linear recurrent sequence over F if there exist $d \in \mathbb{N}$ and $c_j \in F$ for $0 \leq j \leq d$, with $c_d = 1$ such that $\sum_{j=0}^{d} c_j s_{i+j} = 0$,for all $i \in \mathbb{N}$. The polynomial

$$f(X) = \sum_{j=0}^{d} c_j X^j \in F[X]$$

of degree d is called a characteristic polynomial of $\mathcal{S}$.

Moreover, for the linear recurrent sequence $\mathcal{S}$, its minimal polynomial $m_{\mathcal{S}}$ is defined to be the (uniquely determined) characteristic polynomial of $\mathcal{S}$ of least degree.

Definition 4.3.2
The degree of $m_{\mathcal{S}} \in F[X]$ is called the recursion order of $m_{\mathcal{S}}$.

Now we give the next lemma which will be useful in our task.

Lemma 4.3.3 (Lemma 3, [3])
Let $f \in k[X]$ be a polynomial and $\{S_\kappa(f)\}_{\kappa \geq 0}$ be the sequence of the power sums of its roots. Then, that sequence is linearly recurrent and its minimal polynomial is equal to

$$\frac{f}{\gcd(f, f')}.$$

Corollary 4.3.4 (Corollary 3,[3])
Let $f \in k[X]$ be a monic polynomial of degree D over a field of characteristic p. Suppose that all the roots of f have multiplicities less than p. Then f can be computed from the first $2D$ power sums of its roots within $O(M(D)\log(D))$ operations in the base field k.

We present Algorithm 11 for recovering a polynomial by its Newton series. For more information of this algorithm we refer the reader to the proof of Corollary 3 in [3].

Definition 4.3.5
For a rational series $S := \sum_{i \geq 0} s_i t^i \in k[[t]]$, we denote by

$$\mathrm{MinimalPolynomial}(S)$$

the minimal polynomial of the linear recurrent sequence $\mathcal{S} = (s_i)_{i=0}^{\infty}$.

Algorithm 11: Recovering a polynomial by its Newton series (RecoveringPolyFromTraces)

```
Input: The first $2D$ terms of the series $\mathrm{Newton}(f)$.
Output: The polynomial $f$ of degree $D$.
$S_1 \leftarrow \mathrm{Newton}(f) + O(X^{2D})$;
$i \leftarrow 1$;
repeat
    $v_i \leftarrow \mathrm{MinimalPolynomial}(S_i)$;
    $D_i \leftarrow \deg(v_1)$;
    $R_i \leftarrow \mathrm{Newton}(v_i) + O(X^{2D_i})$;
    $S_{i+1} \leftarrow S_i - R_i$;
    $i \leftarrow i + 1$;
until $D_i = 0$;
$\kappa \leftarrow i - 1$;
$v = v_1 v_2 \cdots v_\kappa$;
return $v$;
```

Note that a minimal polynomial of a linear recurrent sequence could be reducible. In our case, the polynomial we are looking for is the minimal polynomial of an element, and is therefore irreducible.

4.4 Minimal Polynomials of Linear Recurrent Sequences

In this section we will describe the algorithm for the computation of the minimal polynomial of a given linear recurrent sequence. This computation plays a vital part in the algorithm for recovering a polynomial by its Newton series. We will use the Berlekamp-Massey algorithm (see [11] for more details) in order to complete this computation.

Let $\mathcal{A} = (a_i)_{i=0}^{\infty}$ over F be a linear recurrent sequence over F. We now indicate how to determine the minimal polynomial $m_{\mathcal{A}}$ of the given sequence $\mathcal{A}$, provided that we know an upper bound $n \in N$ on its recursion order.

Lemma 4.4.1 (Lemma 12.8,[12])
Let $\mathcal{A} = (a_i)_{i=0}^{\infty}$ over F be a linear recurrent sequence over F. Let also $h = \sum_{i \in \mathbb{N}} a_i X^i \in F[[X]]$ be the formal power series whose coefficients are the coefficients of the sequence $\mathcal{A}$, $f \in F[X] \setminus \{0\}$ of degree d and $r = \mathrm{rev}(f)$ its reversal.

1. *The following are equivalent:*
   - *(a) f is a characteristic polynomial of $\mathcal{A}$,*
   - *(b) $r \cdot h$ is a polynomial of degree less than d,*
   - *(c) $h = g/r$ for some $g \in F[X]$ with $\deg(g) < d$.*
2. *If f is the minimal polynomial of $\mathcal{A}$, then $d = \max\{1 + \deg(g), \deg(r)\}$ and $\gcd(g, r) = 1$ in (1.).*

Using the above lemma we can compute the minimal polynomial of a linear recurrent sequence $\mathcal{A} = (a_i)_{i=0}^{\infty}$ over F. Supposing $n \in \mathbb{N}$ is an upper bound on the recursion order of $\mathcal{A}$, then we can compute the minimal polynomial $m_{\mathcal{A}}$ by solving the Padé approximation problem

$$h \equiv \frac{\hat{s}}{\hat{t}} \mod X^{2n}, \quad \text{with} \quad X \nmid \hat{t}, \quad \deg(\hat{s}) < n, \quad \deg(\hat{t}) \leq n, \quad \gcd(\hat{s}, \hat{t}) = 1, \tag{4.3}$$

due to the fact that Lemma 4.4.1(2) yields that $(\hat{s}, \hat{t}) = (g, r)$ is a solution to problem (4.3) (notice that the definition of the reversal implies that $X \nmid r$). We will show that the solution to the Padé approximation problem is unique, up to multiplication by constants, and can be computed by employing the Extended Euclidean Algorithm, using $O(n^2)$ arithmetic operations in F. We will explain the solution to Padé approximation in detail below. Taking it for granted and keeping in mind what we have already studied above lead to the following algorithm for the Minimal polynomial computation of a linear recurrent sequence.

```
Algorithm 12: Minimal Polynomial for $\mathcal{A}$, (Algorithm 12.9, [12])
  Input: An upper bound $n \in \mathbb{N}$ on the recursion order and the first $2n$ entries
         $a_0, a_1, \dots, a_{2n-1} \in F$ of a linear recurrent sequence $\mathcal{A} = (a_i)_{i=0}^{\infty} \in F^{\mathbb{N}}$.
  Output: The minimal polynomial $m_{\mathcal{A}} \in F[X]$ of $\mathcal{A}$.
1 Set $h \leftarrow a_{2n-1}X^{2n-1} + \cdots + a_1X + a_0$;
2 Employ Extended Euclidean Algorithm to compute $\hat{s}, \hat{t} \in F[X]$ such that $\hat{t}(0) = 1$ and the
    Padé approximation problem (4.3) holds;
3 Define $d \leftarrow \max\{1 + \deg(\hat{s}), \deg(\hat{t})\}$;
4 return $\mathrm{rev}(d, \hat{t})$;
```

In order to explain the second step of Algorithm 12, we study the solution to the Padé Approximation Problem (4.3). For a deeper discussion of it, we refer the reader to Section 5.9 in [12].

Padé Approximation (Rational Reconstruction)

Definition 4.4.2
*Let F be a field and $g = \sum_{i \geq 0} g_i X^i \in F[[X]]$ be a formal power series. A **Padé approximant** to g is a rational function $\rho = r/\hat{t} \in F(X)$, with $r, \hat{t} \in F[X]$ and $X \nmid \hat{t}$, that "approximates" g to sufficiently high power of X. Precisely, $r/\hat{t}$ is called $(\kappa, n-\kappa)$-Padé approximant to g if*

$$X \nmid \hat{t} \ \textit{and} \ \frac{r}{\hat{t}} \equiv g \mod X^n, \quad \deg(r) < \kappa, \quad \deg(\hat{t}) \leq n - \kappa. \tag{4.4}$$

In particular, the congruence is equivalent to $r \equiv \hat{t}g \mod X^n$.

Corollary 4.4.3 (Corollary 5.21, Section 5.9, [12])
Let $g \in F[X]$ is of degree κ less than $n \in \mathbb{N}$, namely $0 \leq \kappa \leq n$, and $r_j, s_j, \hat{t}_j \in F[X]$ be the j-th row in the Extended Euclidean Algorithm for $m = X^n$ and g, where j is minimal such that $\deg(r_j) < \kappa$.

1. *There exist polynomials $r, \hat{t} \in F[X]$ satisfying*

$$r \equiv \hat{t}g \mod X^n, \quad \deg(r) < \kappa, \quad \deg(\hat{t}) \leq n - \kappa, \tag{4.5}$$

   namely $r = r_j$ and $\hat{t} = \hat{t}_j$. If in addition $\gcd(r_j, \hat{t}_j) = 1$, then r and $\hat{t}$ also solve (4.4).

2. *If $r/\hat{t} \in F(X)$ is a canonical form solution to (4.4), then $r = \tau^{-1}r_j$ and $\hat{t} = \tau^{-1}\hat{t}_j$, where $\tau \in F \ \ \{0\}$ is the leading coefficient of t_j. In particular, (4.4) is solvable if and only if $\gcd(r_j, \hat{t}_j) = 1$.*

It is worth reminding the Extended Euclidean Algorithm, which is given below. For the sake of simplicity of it, we set up some notation. Let $f \in R = F[X]$ for a field F, where R is a Euclidean domain. We denote by $\mathrm{lc}(f)$ the leading coefficient of f. Then we define $\mathrm{normal}(f) = f/\mathrm{lc}(f)$ as the normal form of f.

Algorithm 13: Extended Euclidean Algorithm (EEA), (Algorithm 3.14, [12])

```
Input: $f, g \in R$, where $R$ is a Euclidean domain with a normal form.
Output: $l \in \mathbb{N}$, $\rho_i, r_i, s_i, \hat{t}_i \in R$ for $0 \leq i \leq l+1$, and $q_i \in R$ for $1 \leq i \leq l$, as computed below.
Set $\rho_0 \leftarrow \mathrm{lc}(f)$,  $r_0 \leftarrow \mathrm{normal}(f) = f/\mathrm{lc}(f)$,  $s_0 \leftarrow \rho_0^{-1}$,  $\hat{t}_0 \leftarrow 0$,
    $\rho_1 \leftarrow \mathrm{lc}(g)$,  $r_1 \leftarrow \mathrm{normal}(g) = g/\mathrm{lc}(g)$,  $s_1 \leftarrow 0$,  $\hat{t}_1 \leftarrow \rho_1^{-1}$ ;
Define $i \leftarrow 1$;
while $r_i \neq 0$ do
    $q_i \leftarrow r_{i-1}$ div $r_i$;
    $\rho_{i+1} \leftarrow \mathrm{lc}(r_{i-1} - q_i r_i)$;
    $r_{i+1} \leftarrow (r_{i-1} - q_i r_i)/\rho_{i+1}$;
    $s_{i+1} \leftarrow (s_{i-1} - q_i s_i)/\rho_{i+1}$;
    $\hat{t}_{i+1} \leftarrow (\hat{t}_{i-1} - q_i \hat{t}_i)/\rho_{i+1}$;
    $i \leftarrow i+1$;
$l \leftarrow i-1$;
return $l$, $\rho_i$ $r_i$, $s_i$, $\hat{t}_i$ for $0 \leq i \leq l+1$, and $q_i$ for $1 \leq i \leq l$;
```

The elements q_i for $1 \leq i \leq l$ are the quotients and the r_i for $0 \leq i \leq l+1$ are the remainders in the Extended Euclidean Algorithm. The elements r_i, s_i, and $\hat{t}_i$ form the i-th row in the Extended Euclidean Algorithm for $0 \leq i \leq l+1$. The property that they fulfill is that $s_i f + \hat{t}_i g = r_i$ for all i. In particular, the elements s_l and $\hat{t}_l$ satisfy that $s_l f + \hat{t}_l g = \gcd(f, g)$.

Here is an example to demonstrate the above corollary.

Example 4.4.4 (Example 5.22, Section 5.9, [12])
Let $g = \sum_{i \geq 0} (i+1)X^i = 1 + 2X + 3X^2 + 4X^3 + \cdots \in \mathbb{F}_5[[X]]$. *Suppose that we want to compute a* $(2,2)$-*Padé approximation to* g. *The Extended Euclidean Algorithm yields*

j	q_j	ρ_j	r_j	s_j	$\hat{t}_j$
0		1	X^4	1	0
1	$X+3$	4	X^3+2X^2+3X+4	0	4
2	X	1	X^2+2X+3	1	$X+3$
3	X^2+2X+3	4	1	X	X^2+3X+1
4		1	0	$4X^3+3X^2+2X+1$	$4X^4$

Thus, we can see in row 3 *that we have*

$$\frac{r_3}{\hat{t}_3} = \frac{1}{X^2+3X+1} = \frac{1}{(X-1)^2}$$

which is the required solution.

Therefore, Corollary 4.4.3 gives a decision-making procedure for Padé approximants, which can be described as follows:

- Compute the appropriate results r_j and $\hat{t}_j$ using Extended Euclidean Algorithm.
- If their gcd is equal to one, then $r_j/\hat{t}_j$ is the unique $(\kappa, n-\kappa)$-Padé approximant as in (4.3). Otherwise, no such approximant exists.

Comment 4.4.5
In particular, the problem (4.3) *is a* (n,n)- *Padé approximation of* h.

As we have thoroughly examined the Padé Approximation Problem, which was the second step of of the Algorithm 12, we can now give a few examples of the computation of the minimal polynomial of a linear recurrent sequence $\mathcal{A}$ using the Algorithm 12.

Example 4.4.6 (Example 12.11 (i), Section 12.3 [12])
Let $F = \mathbb{F}_5$ *and the sequence* $\mathcal{A} = (3, 0, 4, 2, 3, 0, \ldots) \in \mathbb{F}_5^{\mathbb{N}}$ *be linear recurrent of recursion order at most* 3. *Then employing Algorithm* 12 *we have the following results. Step* 1 *yields that*

$$h = 3X^4 + 2X^3 + 4X^2 + 3.$$

By Step 2, *the relevant results of the Extended Euclidean Algorithm for* X^6 *and* h *are given in the next table*

j	q_{j-1}	r_j	$\hat{t}_j$
0		X^6	0
1		$3X^4+2X^3+4X^2+3$	1
2	$2X^2+2X+1$	$4X+2$	$3X^2+3X+4$
3	$2X^3+2X^2$	3	$4X^5+3X^4+X^3+2X^2+1$
4	$3X+4$	0	$3X^6$

By row 2 we read off that

$$h \equiv \frac{4X+2}{3X^2+3X+4} = \frac{X+3}{2X^2+2X+1} \mod X^6.$$

Thus $\hat{s} = X+3$ and $\hat{t} = 2X^2+2X+1$. Finally, in step 3 we have that $d=2$ and thus $m_{\mathcal{A}} = \mathrm{rev}(2,\hat{t}) = X^2+2X+2$. For the sake of completeness, we check that indeed $a_{i+2}+2a_{i+1}+2a_i = 0$ for $i = 0,1,2,3$.

For the sake of completeness, having explained the second step of the Algorithm 12, we collected them and give the aforementioned algorithm under the shape of ready-to-implement pseudo-code.

Algorithm 14: Minimal Polynomial for $\mathcal{A}$, (Algorithm 5.1, [11])

```
Input: An upper bound $n \in \mathbb{N}$ on the recursion order and the first $2n$ entries
       $a_0, a_1, \dots, a_{2n-1} \in F$ of a linear recurrent sequence $\mathcal{A} = (a_i)_{i=0}^{\infty} \in F^{\mathbb{N}}$.
Output: The minimal polynomial $m_{\mathcal{A}} \in F[X]$ of $\mathcal{A}$.
Set $h \leftarrow a_{2n-1}X^{2n-1} + \dots + a_1X + a_0$;
$r_{-1} \leftarrow X^{2n}$ and $r_0 \leftarrow h$;
$\hat{t}_{-1} \leftarrow 0$ and $\hat{t}_0 \leftarrow 1$;
$E_0 \leftarrow 1$ and $F_{-1} = F_0 \leftarrow 1$;
$i \leftarrow 0$;
repeat
    $i \leftarrow i+1$;
    $E_i \leftarrow \mathrm{lc}(r_{i-1})^{\deg(r_{i-2})-\deg(r_{i-1})+1}$ and $F_i \leftarrow E_{i-1}$;
    $q_i \leftarrow E_i r_{i-2}$ div $r_{i-1}$;
    $r_i \leftarrow (E_i r_{i-2} - q_i r_{i-1})/F_i$;
    $\hat{t}_i \leftarrow E_i F_{i-1} \hat{t}_{i-2} - q_i \hat{t}_{i-1}$;
until $\deg(r_i) < n$;
Define $d \leftarrow \max\{1+\deg(r_i), \deg(\hat{t}_i)\}$;
return $\mathrm{rev}(d, \hat{t}_i)$;
```

We proceed now by giving additional details for Algorithm 12 (or Algorithm 14). In particular, we would like to remind that we work over local function fields. We emphasize that in this case the precision plays a vital part for the successful computations over local function fields, and hence for the successful execution of the above algorithm. In more detail, we use in the algorithm that the precision is sufficient to complete the computation of the minimal polynomial of the linear recurrent sequence. Performing the Extended Euclidean Algorithm as many times as it needed for the accomplishment of the second step of the Algorithm 12 results in precision losses. Consequently, this can lead to a minimal polynomial with coefficients of very small precision which is not a reliable result as it can be a polynomial without a constant term. With the intention of avoiding such a case, we increase the precision in advance before carrying out the algorithm steps.

The Extended Euclidean Algorithm 13 for polynomials $f, g \in F[X]$ with $\deg(f) = n \geq \deg(g) = m$ can be performed with at most $m+2$ inversions (see Theorem 3.16, [12]). In our case we have $f(X) = X^{2N} \in k[X]$ and $g = h \in k[X]$ with $\deg g = 2N-1$, and hence the Extended Euclidean Algorithm 13 can be performed with at most $2N+1$ inversions.

Experiments drive us getting the following heuristic result. We set the precision by the following formula

$$\mathrm{prec} := \max(\mathcal{P}_{\mathcal{A}}) + v_F(a_{2N-1}) \cdot N,$$

where $\mathcal{P}_{\mathcal{A}}$ is a list with the F-precision of a_i, namely

$$\mathcal{P}_{\mathcal{A}} := [\mathrm{prec}_F(a_\kappa) \ : \ 0 \leq \kappa \leq 2N-1].$$

Let us now describe our case. Let us assume that $f \in k[X]$, where $k := \mathbb{F}_q((t))$ be an irreducible polynomial of degree $N = \deg(f)$ with $f(\alpha) = 0$. Let also $\beta = \sum_{i \geq 0} b_i \alpha^i$ and $\tilde{\beta} = \frac{1}{t^d}\beta$. Given an upper bound $N \in \mathbb{N}$ on the recursion order and the first $2N$ entries $\{S_\kappa = S_\kappa(\tilde{\beta}) \in k\}_{\kappa \geq 0}^{2N-1}$ of a linear recurrent sequence $\mathcal{S} = (S_i)_{i=0}^{\infty} \in k^{\mathbb{N}}$ of the higher traces of $\tilde{\beta}$, we would like to compute the minimal polynomial $m_{\mathcal{S}} \in k[X]$ of $\mathcal{S}$. Write $\mathcal{P}_S$ to be a list with the k-precision of the higher traces of $\tilde{\beta}$, namely

$$\mathcal{P}_S := [\mathrm{prec}_k(S_\kappa(\tilde{\beta})) \ : \ 0 \leq \kappa \leq 2N-1],$$

and then define

$$\mathrm{prec} := \max(\mathcal{P}_S) + v_k(S_{2N-1}) \cdot N.$$

Then, we update the base field k to the new precision and set

$$h \leftarrow S_{2n-1}X^{2n-1} + \cdots + S_1 X + S_0 \in k[X].$$

After that, we proceed exactly as Algorithm 14 describes.

It is worth pointing out that after the computation of the minimal polynomial $m_{\mathcal{S}} \in k[X]$, we change the precision of k back to the one we started with.

Another point we would like to address is that since during the steps of the Extended Euclidean Algorithm, (precisely, the repeat-code of the Algorithm 14), we work with polynomials with coefficients of some precision, we should set a bound which will be the precision beyond it an element will be considered as zero. Let us denote this bound by $\tilde{O}_0$. Then we use $\tilde{O}_0$ as a criterion in the polynomials r_i of the repeat-code of the Algorithm 14. Heuristically, we set

$$\tilde{O}_0 := \max(\mathcal{P}_S) + \min(\mathcal{P}_S).$$

4.5 Defining Polynomials of Extensions

Taking into account all the theory analyzed above, we can now develop our approach for the computation of the generating polynomial of an extension.

For a start, we establish some notation. Let $f \in k[X]$ be a polynomial of degree n and $f(\alpha) = 0$. Let also $\{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ be the n distinct k-embeddings of $k(\alpha)$ into an algebraic closure $\bar{k}$ of k and $S := \mathrm{Tr}_{k(\alpha)/k}$ stands for the trace of $k(\alpha)/k$. We would like to compute the minimal polynomial of the element $\tilde{\beta} = \frac{1}{\pi_k^d}\beta$, where $\beta = \sum_{i=0}^{n-1} a_i \alpha^i$ and $\pi_k = t$.

We can easily check that

$$S_\kappa(\alpha^j) = S_{\kappa j}(\alpha)$$

and

$$S_\kappa(\beta) = \sum_{i=1}^{n} \sigma_i(\beta)^\kappa = \sum_{i=1}^{n} \sigma_i(\beta^\kappa) = S(\beta^\kappa) = \mathrm{Tr}_{k(\alpha)/k}(\beta^\kappa). \tag{4.6}$$

In addition, let $\beta^\kappa = \sum_{j=0}^{n-1} a_{\kappa,j}\alpha^j$, then

$$S_\kappa(\beta) = S(\beta^\kappa) = \sum_{j=0}^{n-1} a_{\kappa,j} S(\alpha^j) = \sum_{j=0}^{n-1} a_{\kappa,j} S_j(\alpha). \tag{4.7}$$

With the purpose of computing the minimal polynomial of $\tilde{\beta}$ we can follow two different strategies, which we will separately develop in the next two subsections.

4.5.1 Computation without Denominators

In this subsection we study the case of computing the minimal polynomial of $\tilde{\beta}$ by using the minimal polynomial of β and making a suitable transformation in order to achieve the desired minimal polynomial.

To be more precise, given that $\tilde{\beta} = \frac{1}{t^{\tilde{d}}}\beta$, we can see that

$$m_{\tilde{\beta}}(X) = \frac{m_\beta(t^{\tilde{d}}X)}{t^{n\tilde{d}}}. \tag{4.8}$$

Therefore, taking into consideration the above formula, we can compute the minimal polynomial of $\tilde{\beta}$ by computing the minimal polynomial of β and using the above formula (4.8). Thus, now we have reduced the problem to the computation of m_β.

It is worth bearing in mind that for the computation of m_β we can easily handle some special cases of β, which are analyzed below.

Special Cases:

- If $\beta \in k$, where $k = \mathbb{F}_q((t))$ or $k = \mathbb{F}_{q_i}((u_i))$, then $m_\beta = X - \beta$.
- If $\beta = \alpha$, then $m_\beta = f$.
- If β is linear, that is to say $\beta = b_0 + b_1\alpha$, then $m_\beta = m_\alpha(\frac{X-b_0}{b_1})$.

Another case we can deal with is when $\beta = \alpha^\ell$ with $\ell \geq 2$, and this can be done by using the next theorem.

Theorem 4.5.1 (Theorem 7, [25])
Let A be a polynomial of positive degree m over an integral domain R with roots $\alpha_1, \cdots, \alpha_m$. Let p, q be positive integers. Then

$$r(X) = \mathrm{res}_Y(A(Y), X^q - Y^p)$$

has the roots $\alpha_i^{p/q}$ with $i := 1, \cdots, m$.

According to the above theorem, if $\beta = \alpha^\ell$ with $\ell \geq 2$, then

$$m_\beta(X) = \mathrm{res}_Y(m_\alpha(Y), X - Y^\ell).$$

Now we can present the algorithm of the m_β computation.

Algorithm 15: Minimal Polynomial for β (chi_small_char)

```
Input: A polynomial $f \in k[X]$ of $\deg(f) = n \in \mathbb{N}$ with $f(\alpha) = 0$, the first $n$ higher traces
       $S_0(\alpha), S_1(\alpha), \dots, S_{n-1}(\alpha) \in k$ of $f$ and the element $\beta$.
Output: The minimal polynomial of $\beta$.
if $\beta$ belongs to special cases then return $m_\beta$ as defined above;
Compute $\mathcal{S} = [S_0(\beta), S_1(\beta), \dots, S_{2n-1}(\beta)] = [\mathrm{Tr}_{k(\alpha)/k}(1), \mathrm{Tr}_{k(\alpha)/k}(\beta), \dots, \mathrm{Tr}_{k(\alpha)/k}(\beta^{2n-1})]$
  using (4.7);
Recover the polynomial by its higher traces $\mathcal{S}$ by employing Algorithm 11. That is,
  $m_\beta \longleftarrow$ RecoveringPolyFromTraces($\mathcal{S}$);
Check $m_\beta$, that is CheckChiPoly($m_\beta, \beta, f$);
return $m_\beta$;
```

What is left now is to outline how we check that the resulting minimal polynomial fulfills the following property

$$m_\beta(\beta) = 0.$$

Before terminating the algorithm, we check if that property is fulfilled by employing the algorithm called "CheckChiPoly". If $m_\beta(\beta) = 0$ is fulfilled, then we return m_β. Otherwise, we set

$$\mathrm{prec} := \min\{v_k(c_i) \;:\; 0 \leq i \leq n-1\},$$

where $m_\beta(\beta) = \sum_{i=0}^{n-1} c_i\alpha^i \in k(\alpha) = k[X]/\langle f \rangle$. We update the precision of the coefficient field of m_β to the new one. This means that we reduce digits from the polynomial m_β. Then, we check again if the "new" truncated minimal polynomial fulfills that $m_\beta(\beta) = 0$. Similarly, we repeat the same step until we cancel out all the digits needed in order for β to be a root of m_β. Finally, the updated m_β is returned.

Comment 4.5.2
It should be noted that the above described method works in the first attempt. That is to say, when the check of

$$m_\beta(\beta) = 0$$

is failed and we find the new precision, called prec, *then* prec *is the desired one. In other words, it guarantees that the defining the minimal polynomial m_β with precision equal to* prec, *then $m_\beta(\beta) = 0$.*

The corresponding algorithm is summarized below.

Algorithm 16: Check m_β (CheckChiPoly)

```
Input: A polynomial $f \in k[X]$ of $\deg(f) = n \in \mathbb{N}$, the element $\beta$ and its minimal polynomial $m_\beta \in k[X]$.
Output: Return $m_\beta$ such that $m_\beta(\beta) = 0$.
prec $\longleftarrow$ prec($k$);
while $m_\beta(\beta) \neq 0$ do
    Update the precision of $k$ to prec;
    Define $k(\alpha) = k[X]/\langle f \rangle$;
    Compute $m_\beta(\beta)$ in $k(\alpha)$;
    if $m_\beta(\beta) = 0$ then
        return $m_\beta$;
    else
        Set prec $\longleftarrow \min\{v_k(c_i) \ : \ 0 \leq i \leq n-1\}$, where $m_\beta(\beta) = \sum_{i=0}^{n-1} c_i \alpha^i$;
        Define $k$ with the new prec and $m_\beta \leftarrow m_\beta \in k[X]$;
return $m_\beta$;
```

4.5.2 Computation with Denominators

In this subsection we address the case of computing the minimal polynomial of $\tilde{\beta}$ by taking into account the denominator of $\tilde{\beta}$.

We assume that $\tilde{\beta} = \frac{1}{t^{\tilde{d}}}\beta$ and we know that

$$m_{\tilde{\beta}}(X) = \frac{m_\beta(t^{\tilde{d}}X)}{t^{n\tilde{d}}}. \tag{4.9}$$

Nevertheless, using (4.9) is slower than the direct computation of $m_{\tilde{\beta}}$. Moreover, we can easily get that

$$S_\kappa(\tilde{\beta}) = \frac{S_\kappa(\beta)}{t^{\tilde{d}\kappa}},$$

where $\kappa \geq 0$. It is important to note that $\beta \in \mathcal{O}_k[\alpha]$ and $\tilde{\beta} \in \mathcal{O}_k \setminus \mathcal{O}_k[\alpha]$, and so $S_\kappa(\tilde{\beta}) \in \mathcal{O}_k$ for every $\kappa \geq 0$.

Taking advantage of the above formula of the higher traces, we can directly use the higher traces of $\tilde{\beta}$ during the execution of the minimal polynomial algorithm. In other words, we will recover the polynomial formed by $\{S_\kappa(\tilde{\beta})\}_{\kappa=0}^{2n-1}$. In particular, after the computation of

$$\mathcal{S} = [S_0(\beta), S_1(\beta), \dots, S_{2n-1}(\beta)],$$

by using (4.7), we determine the sequence

$$\mathcal{S} = [S_0(\tilde{\beta}), S_1(\tilde{\beta}), \dots, S_{2n-1}(\tilde{\beta})] = [S_0(\beta), \frac{S_1(\beta)}{t^{\tilde{d}}}, \dots, \frac{S_{2n-1}(\beta)}{t^{\tilde{d}(2n-1)}}]$$

and then we recover the polynomial by its higher traces $\mathcal{S}$ by employing Algorithm 11. That is,

$$m_{\tilde{\beta}} \longleftarrow \text{RecoveringPolyFromTraces}(\mathcal{S}).$$

Thus, after adding this step to Algorithm 15, it will directly return $m_{\tilde{\beta}}$.

As a result, the use of denominators reduces the precision, and thus the computations would be done in a more efficient way, as we avoid computing unnecessary information.

It is important to point out the following remark about the minimal polynomial algorithm, which applies to both strategies.

Remark 4.5.3
Having computed a sequence $\mathcal{S}$ of higher traces, we would like to recover the polynomial formed by this sequence. To do so, we employ Algorithm 11, which is called RecoveringPolyFromTraces($\mathcal{S}$)*. It is worth bearing in mind that this algorithm determines the desired polynomial in the first loop of its execution steps. This is due to the fact that the polynomial we are looking for is the defining polynomial of the extension, and is therefore irreducible.*

4.6 Examples

For the sake of completeness we list a table of examples in which we compare the time required for the factorization of polynomials using the "Extensions"-parameter when employing Magma's factoring algorithm and our algorithm, respectively. This way, we can check the performance of the computation of the defining polynomial of the corresponding extension. This table can be found in Section A.4.

We note that in cases of large precision our algorithm performs faster than Magma's algorithm as indicated in Table A.16.

Chapter 5

Splitting Fields: Combined Approach

5.1 Splitting Fields in the General Case

In this section we describe an improved approach of computing the splitting field of a given polynomial. To be more precise, in this approach we combine our approach developed in the section "A Variation of the Basic Approach" (see Section 3.2) with the one developed by Greve in his thesis [15], which we studied in the previous sections (see Sections 1.9, 2.3.1, 2.4).

It is worth reminding that the key idea of our approach of computing the splitting field of a polynomial over a local function field is that every local function field with finite residue field $\mathbb{F}_{\tilde{q}}$ is isomorphic to the field of formal Laurent series $\mathbb{F}_{\tilde{q}}((u))$. Especially, during the steps of the splitting field computation we work over fields of the form $\mathbb{F}_{\tilde{q}}((u))$ instead of working over an extension of $K := \mathbb{F}_q((t))$ by adjoining a root α of the initial polynomial.

On the other hand, the vital part of Greve's method is the computation of a subfield T of the splitting field N of an Eisenstein polynomial $f \in K[X]$. Particularly, the extension T is the maximal tamely ramified subextension of N/K, i.e. N is a p-extension of T. This implies that $\mathrm{Gal}(N/T)$ is a p-group, where p is the characteristic of K. This is summarized in Theorem 2.4.1.

It should be noted that the ramification polygon does not provide enough information to completely theoretically describe the splitting field of an Eisenstein polynomial. Nevertheless, by computing the field T, it can speed up the determination of the splitting field for many polynomials.

Eisenstein Polynomials

For a start, let us describe our approach if the initial polynomial $f \in K[X]$ is an Eisenstein polynomial. We begin with the determination of the subfield T using Theorem 2.4.1. To do so, we use the Algorithm 6. After the execution of the latter algorithm, we have that T is an extension of K. However, we know that it is isomorphic to a local function field. Therefore, we can specify it, which is needed in order to proceed with the computation of the splitting field. This is why we have extended the Algorithm 6 to one that the desired field T has been transformed to a formal Laurent series field, which is given in the next algorithm.

```
Algorithm 17: $p$-Reduction LFF [$p$—ReductionLFF]
  Input: An Eisenstein polynomial $f(X) = \sum_{i=0}^{n} a_i X^i \in \mathcal{O}_K[X]$ of degree $n = e_0 p^m$, with $p \nmid e_0$ and
         $K = \mathbb{F}_p((t))$
  Output: A local function field $T$ over $K$ such that the splitting field of $f$ ia a $p$-extension over $T$.
  Determine the field $T$ using Algorithm 6, that is $T \leftarrow p$-Reduction($f$);
  Compute $T \cong \mathbb{F}_{\tilde{q}}((u_j))$;
  return $T \longleftarrow \mathbb{F}_{\tilde{q}}((u_j))$;
```

Having computed the field $T \cong \mathbb{F}_{\tilde{q}}((u_j))$, we upgrade the polynomial $f \in K[X]$ to a polynomial over T, i.e. $f \in T[X]$, as it holds that

$$\mathrm{Spl}(f, K) \cong \mathrm{Spl}(f, T).$$

Then we proceed by computing the splitting field of $f \in T[X]$, employing our so-called algorithm "SplittingFieldLFF", (see Algorithm 9, Chapter 3). Precisely, we repeat the same procedure as it has been described in the latter algorithm for the determination of the splitting field of $f \in T[X]$. Therefore, we end up computing the desired splitting field of $f \in K[X]$.

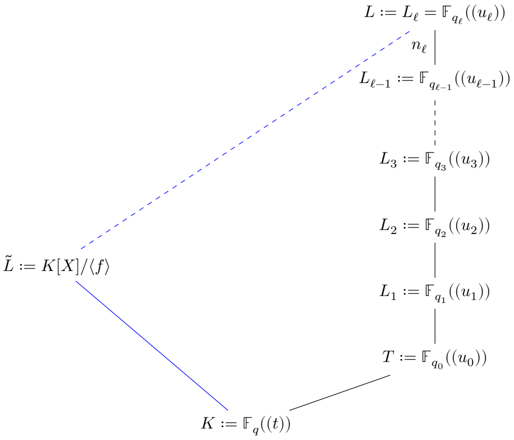

$L := L_\ell = \mathbb{F}_{q_\ell}((u_\ell))$
n_ℓ
$L_{\ell-1} := \mathbb{F}_{q_{\ell-1}}((u_{\ell-1}))$
$L_3 := \mathbb{F}_{q_3}((u_3))$
$L_2 := \mathbb{F}_{q_2}((u_2))$
$\tilde{L} := K[X]/\langle f \rangle$
$L_1 := \mathbb{F}_{q_1}((u_1))$
$T := \mathbb{F}_{q_0}((u_0))$
$K := \mathbb{F}_q((t))$

Figure 5.1: **Splitting Field: Combined Approach**

It is worth pointing out that in this approach for the splitting field computation we have to make some adjustments to our precision determination approach analyzed in Chapter 3 (see Heuristic Assumptions 3.2.10). Experiments drive us getting a similar heuristic result.

Heuristic Assumption 5.1.1

In every step i we increase the u_i-precision in the following way:

- *If $i = 1$, then we set $v_{L_1}(\alpha) := e(T/K)$ and $O_1 := (e(L_1/T) + e(T/K) - 1)M_1 + 50$.*
- *If $i > 1$, then we set $v_{L_i}(\alpha) := v_{L_{i-1}}(\alpha)e(L_i/L_{i-1})$ and $O_i := e(L_i/K)M_1 + 20$.*

Nevertheless, experiments showed that the determined precision was not enough to achieve the factorization in some examples. In order to solve this issue, we can determine the precision by adding some extra conditions, which is given in the next result. We keep all the notations defined in Heuristic Assumption 5.1.1.

Heuristic Assumption 5.1.2

Under the same conditions as above and let $M_{O_{\hat{g}_i}} = \max\{\mathrm{Prec}(\tilde{a}_j) : 0 \leq j \leq n_1\}$, where $\hat{g}_i = \sum_{j=0}^{n_1} \tilde{a}_j X^j$ is the corresponding Eisenstein polynomial of $g_i = \min(L_i/L_{i-1})$. Having determined the precision by Heuristic Assumption 5.1.1, we also consider the following:
For $i = 1$:

- $O_1 := \begin{cases} O_1 + M_f M_1 & \textit{if } M_f > 5 \\ O_1 & \textit{else} \end{cases}$ *, where $M_f := \max\{v_t(a_j) : 0 \leq j \leq n\}$ and also*
- $O_1 := \begin{cases} M_{O_{\hat{g}_1}} + O_1 + 20 & \textit{if } O_1 < M_{O_{\hat{g}_1}} \,\&\, M_f > 3 \,\&\, [T : K] > 1 \\ O_1 & \textit{else} \end{cases}$.

For $i > 1$: $O_i := \begin{cases} M_{O_{\hat{g}_i}} + e(L_i/K) + 10 & \textit{if } O_i < M_{O_{\hat{g}_i}} \\ O_i & \textit{else} \end{cases}$.

On the whole, the splitting field algorithm can be summarized in an algorithmic form, which is given below.

Algorithm 18: Splitting Field over LFF [SplittingFieldLFF_ANA]

```
Input: An Eisenstein polynomial $f \in \mathcal{O}_K[X]$.
Output: The splitting field of $f$.

Compute the ramification polygon of $f$;
Determine the field $T$, that is $T \leftarrow p$-ReductionLFF$(f)$;
Upgrade $f$ over the field $T$, that is $\hat{f} \longleftarrow f(u_j, X) \in T[X]$;
Repeat the steps of the splitting field algorithm SplittingFieldLFF for $\hat{f}$, and then,
        $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} \leftarrow \mathrm{SplittingFieldLFF}(\hat{f})$
    where Spl$(f)$ is the splitting field of $f$ over $K$, rSF is a list of the roots of $f$, subst is a list with the
    substitutions that we make of f in order to compute the splitting field and data is a dataset with the
    minimal polynomial in each intermediate field extension;
return $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data}$;
```

Irreducible & non-Eisenstein Polynomials

Given an Eisenstein polynomial we can compute its splitting field by using Theorem 2.4.1 or Algorithm 6. According to Theorem 2.4.1, this method can not be applied in case of irreducible non-Eisenstein polynomials. However, we would like to study further those cases and develop an algorithm for attacking them as well.

Let us now suppose that $f \in K[X]$ of degree n, where $K = \mathbb{F}_q((t))$, is an irreducible non-Eisenstein polynomial. We will analyze our methodology for dealing with such a case.

For a fuller treatment of such a case, we can divide it into three sub-cases:

- Totally Ramified Case, namely $e = \deg(f)$ & $\mathfrak{f} = 1$,
- Totally Unramified Case, namely $\mathfrak{f} = \deg(f)$ & $e = 1$,
- Mixed Case, namely $e > 1$ & $\mathfrak{f} > 1$,

which will be thoroughly examined.

Totally Ramified Case, namely $e = \deg(f)$ & $\mathfrak{f} = 1$:

We begin with the case of totally ramified. Using the factorization code, we can find an Eisenstein polynomial $g \in K[X]$ which generates the same extension as f. That is to say,

$$K[X]/\langle f(X)\rangle \cong K[X]/\langle g(X)\rangle.$$

Moreover, we can determine its uniformizing element π_K such that $g(\pi_K) = 0$. Then, we compute the splitting field of the corresponding Eisenstein polynomial g, since

$$N := \mathrm{Spl}(f) \cong \mathrm{Spl}(g) \quad \text{and}$$
$$\mathrm{Gal}(f|K) \cong \mathrm{Gal}(g|K).$$

Thus, we have reduced the problem to the one of the Eisenstein polynomials.

It is important to point out that although by performing the above strategy we do compute the Spl(f), we do not have access to its roots as we have computed the roots of g. Nevertheless, it is easy to fix it as we can specify the roots of f by using the roots of g.

We now describe how we can determine the roots of f. To do so, the key ingredient is the uniformizing element π_K. In particular, we have access to a relation of π_K and the roots of f, that is to say

$$\pi_K = A(\alpha), \text{ such that } A \in K[X], \ f(\alpha) = 0. \tag{5.1}$$

We would like to find $B \in K[X]$ such that

$$\alpha = B(\pi_K). \tag{5.2}$$

So then

$$\alpha = B(\pi_K) \overset{(5.1)}{\Rightarrow} \alpha = B(A(\alpha))$$
$$\Rightarrow X = B(A(X)) \mod f.$$

Assume that $B(X) = \sum_{i=0}^{n-1} b_i X^i$. Then we have that

$$B(A(X)) = \sum_{i=0}^{n-1} b_i A(X)^i \tag{5.3}$$

and let denote $A_i := A(X)^i \mod f$. This yields that $A_0, \dots, A_{n-1} \in K[X]$ of degree less than or equal to $n-1$. Therefore, considering the equation (5.3) modulo f we have that:

$$B(A(X)) \equiv \sum_{i=0}^{n-1} b_i A(X)^i = \sum_{i=0}^{n-1} b_i A_i \mod f$$

and so

$$\sum_{i=0}^{n-1} b_i A_i = X,$$

which is a system of n equations. Thus, solving the above system of equations, we specify b_i for each i, and so B is determined. As a result, using the formula B, we can determine the roots of f.

Proposition 5.1.3
Let $f \in K[X]$ of degree n, where $K = \mathbb{F}_q((t))$, be an irreducible totally ramified non-Eisenstein polynomial, and $g \in K[X]$ be the corresponding Eisenstein polynomial which generates the same extension as f, as above. Let also π_K, B and A as defined above, (see (5.1) and (5.2)). Then it holds that every $\sigma \in \mathrm{Gal}(g|K) \cong \mathrm{Gal}(N/K)$ imposes an automorphism $\tau \in \mathrm{Gal}(f|K) \cong \mathrm{Gal}(N/K)$.

Proof. The equation $\pi_K = A(\alpha)$,(see (5.1)), yields that $K[X]/\langle g\rangle \leqslant K[X]/\langle f\rangle$ and since f is irreducible and $\deg(f) = \deg(g)$, it holds that $K[X]/\langle g\rangle = K[X]/\langle f\rangle$. Thus, f and g have the same splitting field, that is $N = \mathrm{Spl}(f) \cong \mathrm{Spl}(g)$. Let now $\sigma \in \mathrm{Aut}(N/K)$. Then σ induces a permutation on the roots of f, but also does on the roots of g, which completes the proof. □

What is left is to show that the formula B, (see (5.2)), gives the roots of f.

Proposition 5.1.4
Let $f \in K[X]$ of degree n, where $K = \mathbb{F}_q((t))$, be an irreducible totally ramified non-Eisenstein polynomial, and $g \in K[X]$ be the corresponding Eisenstein polynomial which generates the same extension as f, as above. Let also π_K, B and A as defined above, (see (5.1) and (5.2)), and $N := K(\pi_1, \dots, \pi_n) = K(\alpha_1, \dots, \alpha_n)$, where $\pi_1, \dots, \pi_n$ and $\alpha_1, \dots, \alpha_n$ denote the roots of g and f respectively. Then it holds that

$$\alpha_i = B(\pi_j)$$

for every $1 \leq i \leq n$ and some j such that $1 \leq j \leq n$.

Proof. Let $\pi_1 := \pi_K = A(\alpha) := A(\alpha_1)$. As we explained above, it yields to the relation

$$\alpha_1 = B(\pi_1). \tag{5.4}$$

Define $\hat{g}(X) := f(B(X)) \in K[X]$. Then,

$$\hat{g}(\pi_1) = f(B(\pi_1)) \overset{(5.4)}{=} f(\alpha_1) = 0.$$

Since g is the minimal polynomial of π_1 over K, we get $g \mid \hat{g}$, which means that for every root π_i of g we get that

$$g(\pi_i) = 0 \Rightarrow \hat{g}(\pi_i) = 0,$$

and so $f(B(\pi_i)) = 0$. Therefore, $B(\pi_i)$ is a root of f for every $1 \leq i \leq n$.
The only point remaining concerns that for every $1 \leq i \neq j \leq n$ such that $\alpha_i = B(\pi_\lambda)$ and $\alpha_j = B(\pi_\mu)$, with $1 \leq \lambda, \mu \leq n$, then $\lambda \neq \mu$, which means $\pi_\lambda \neq \pi_\mu$. There are $\tau_i \neq \tau_j \in \mathrm{Aut}(N/K)$ such that $\tau_i(\alpha_1) = \alpha_i$, $\tau_j(\alpha_1) = \alpha_j$ and without loss of generality $\tau_i(\pi_1) = \pi_i$, $\tau_j(\pi_1) = \pi_j$, by Proposition 5.1.3. Then

$$\alpha_i = \tau_i(\alpha_1) = \tau_i(B(\pi_1)) = B(\tau_i(\pi_1)) = B(\pi_i).$$

Similarly, we get $\alpha_j = B(\pi_j)$, and so $\pi_j \neq \pi_j$ as required. □

For the better understanding we give an example.

Example 5.1.5
Let $f(X) = X^3 + tX^2 + t^2 \in K[X]$, where $K := \mathbb{F}_3((t))$ and $f(\alpha) = 0$. Obviously, this polynomial is not Eisenstein and using the Factorization code we can find that $e = 3$, $\mathfrak{f} = 1$ and $E \cong K[X]/\langle g(X)\rangle$, where $g(X) = X^3 + 2tX^2 + tX + 2t \in K[X]$ is Eisenstein and its primitive element $\pi_K = \frac{\alpha}{t}$ such that $g(\pi_K) = 0$.

By employing the Splitting Field algorithm of Eisenstein polynomials for g, we find that

$$N = \mathrm{Spl}(g) = \mathrm{Spl}(f) = \mathbb{F}_3((u_2)), \text{ with } \mathrm{prec} = 62$$

where $\Phi_1(t) = 2u_1^2 \mod u_1^{62}$ *and*
$\Phi_2(u_1) = u_2^3+u_2^5+u_2^9+2u_2^{11}+u_2^{17}+u_2^{27}+2u_2^{29}+2u_2^{35}+u_2^{53} \mod u_2^{62}$. *Also, we have that* $[N : K] = 6$ *and the roots of* g *are*

$$\begin{aligned} r_1 &:= 2u_2^2 + u_2^4 + 2u_2^6 + u_2^{10} + u_2^{12} + 2u_2^{18} + u_2^{28} + u_2^{30} + u_2^{36} + 2u_2^{54} \mod u_2^{58},\\ r_2 &:= 2u_2^2 + 2u_2^3 + u_2^4 + u_2^5 + 2u_2^7 + 2u_2^8 + 2u_2^9 + 2u_2^{10} + 2u_2^{11} + u_2^{16} + u_2^{17} + 2u_2^{25} + 2u_2^{26}\\ &\qquad + 2u_2^{27} + 2u_2^{28} + 2u_2^{29} + 2u_2^{34} + 2u_2^{35} + u_2^{52} + u_2^{53} \mod u_2^{58},\\ r_3 &:= 2u_2^2 + u_2^3 + u_2^4 + 2u_2^5 + u_2^7 + 2u_2^8 + u_2^9 + 2u_2^{10} + u_2^{11} + u_2^{16} + 2u_2^{17} + u_2^{25} + 2u_2^{26} + u_2^{27}\\ &\qquad + 2u2^{28} + u_2^{29} + 2u_2^{34} + u_2^{35} + u_2^{52} + 2u_2^{53} \mod u_2^{58}. \end{aligned}$$

Having access to the roots of g*, we would like to determine the roots of* f*. So we want to find the transformation formula. In this case, we have that* $A(X) = \frac{X}{t}$*. Assume that* $B(X) = \sum_{i=0}^{n-1} b_i X^i = b_0 + b_1X + b_2X^2$*. Then, we have that*

$$B(A(x)) = b_0 + b_1A(X) + b_2A(X)^2 = b_0 + b_1\frac{X^2}{t} + b_2\frac{X^4}{t^2} \tag{5.5}$$

Moreover, $f(X) = 0 \Rightarrow X^3 + tX^2 + t^2 = 0 \Rightarrow X^3 = -tX^2 - t^2$*, and then* $X^4 = tX^3 - t^2X$*. Thus, considering the equation* (5.3) *modulo* f *yields that*

$$B(A(X)) = b_0 + b_1\frac{X^2}{t} - b_2\frac{X^3}{t} - b_2X.$$

However, we know that

$$\begin{aligned} B(A(X)) = X \quad &\Rightarrow \quad b_0 + b_1\frac{X^2}{t} - b_2\frac{X^3}{t} - b_2X = X \Rightarrow b_0 + b_1\frac{X^2}{t} - \frac{b_2}{t}(-tX^2 - t^2) - b_2X = X\\ &\Rightarrow \quad b_0 + b_1\frac{X^2}{t} + b_2X^2 + b_2t - b_2X = X \Rightarrow (\frac{b_1}{t} + b_2)X^2 - b_2X + (b_0 + b_2t) = X \end{aligned}$$

So we have that

$$\begin{aligned} &\frac{b_1}{t} + b_2 = 0\\ &-b_2 = 1\\ &b_0 + b_2t = 0. \end{aligned}$$

Solving the above system of equation we have that $b_0 = t,\ b_1 = t,\ b_2 = -1$*. Therefore,* $B(X) = t + tX - X^2$*, and so* $\alpha = B(\pi_K) = t + t\pi_K - \pi_K^2$*.*
Using the embedded elements t *and* u_1 *we consider* $B(X) \in N[X]$*, and substituting* r_i *for every* $i = 1, 2, 3$ *we end up finding the roots of* f*, which are given below:*

$$R := [B(r_1), B(r_2), B(r_3)] = [\rho_1, \rho_2, \rho_3],$$

where

$$\begin{aligned} \rho_1 &:= 2u_2^4 + u_2^6 + 2u_2^8 + 2u_2^{10} + 2u_2^{12} + u_2^{18} + 2u_2^{26} + 2u_2^{28} + 2u_2^{30} + 2u_2^{36} + u_2^{54} \mod u_2^{60},\\ \rho_2 &:= 2u_2^4 + u_2^5 + u_2^7 + u_2^{10} + 2u_2^{11} + 2u_2^{16} + u_2^{17} + u_2^{25} + u_2^{28} + 2u_2^{29} + u_2^{34} + 2u_2^{35}\\ &\quad + 2u_2^{52} + u_2^{53} \mod u_2^{60},\\ \rho_3 &:= 2u_2^4 + 2u_2^5 + 2u_2^7 + u_2^{10} + u_2^{11} + 2u_2^{16} + 2u_2^{17} + 2u_2^{25} + u_2^{28} + u_2^{29} + u_2^{34} + u_2^{35}\\ &\quad + 2u_2^{52} + 2u_2^{53} \mod u_2^{60}. \end{aligned}$$

This is summarized in the next algorithm:

Algorithm 19: Splitting Field Totally Ramified [SplittingFieldLFF_ANA_TR]

```
Input: A non-Eisenstein polynomial $f \in \mathcal{O}_K[X]$.
Output: The Splitting Field of $f$.
Determine the corresponding Eisenstein polynomial $g \in K[X]$ such that $K[X]/\langle f(X)\rangle \cong K[X]/\langle g(X)\rangle$ and its primitive element $\pi_K$ such that $g(\pi_K) = 0$;
Employ the splitting field algorithm for Eisenstein polynomial SplittingFieldLFF_ANA for $g$, and then,
    $\mathrm{Spl}(g), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} := \mathrm{SplittingFieldLFF\_ANA}(g)$
  where $\mathrm{Spl}(g) = \mathrm{Spl}(f)$ is the splitting field of $f$ over $K$, rSF is a list of the roots of $g$, subst is a list with the substitutions that we make of f in order to compute the splitting field and data is a dataset with the minimal polynomial in each intermediate field extension;
Find the transformation formula $B(X) = \sum_{i=0}^{n-1} b_i X^i$ such that $\alpha = B(\pi_K)$;
Compute the roots of $f$, R $= [B(\rho) \ : \ \rho \in \mathrm{rSF}]$;
return $\mathrm{Spl}(f), \mathrm{size}, \mathrm{R}, \mathrm{subst}, \mathrm{data}$;
```

Totally Unramified Case, namely $e = 1$ & $\mathfrak{f} = \deg(f)$:

Let us now analyze this situation. First of all, we construct the unramified extension, which in case of local function fields is a constant field extension. Consequently, we have that

$$U \cong \mathbb{F}_{q^{\mathfrak{f}}}((u_1)).$$

Then, we consider f over U, namely $f \in U[X]$ and factorize it. After factorizing it, f splits in linear factors. In other words, this method is what we do in the algorithm called SplittingFieldLFF. Especially, this algorithm gives the result in the first step for this kind of polynomials.

Mixed Case, namely $e > 1$ & $\mathfrak{f} > 1$:

Finally, we study the mixed case in which we have non-trivial ramification index and inertia degree. By using the factorization code, we find an extension generated by f, and let denote it by E. In this case, firstly, we construct the unramified extension, which is a constant field extension of the following form $U := \mathbb{F}_{q^{\mathfrak{f}}}((u_1))$.

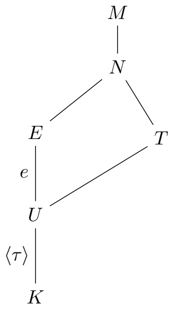


Figure 5.2: **Mixed Case**

Having constructed the extension U/K, we want to compute the field T by using the Eisenstein polynomial f_{Eis} provided by the extension E.

Consider f_{Eis} over U. Then define the Frobenius automorphism

$$\tau : \; U \to U, \;\; \tau(\omega) = \omega^q \text{ and } \tau(u_1) = u_1.$$

Thus, now we define the list below

$$\text{NonLinFact} := [\tau^i(f_{\text{Eis}}), \; 0 \leq i \leq d-1],$$

which includes the factors of f, since we know that

$$f = \prod_{i=0}^{d-1} \tau^i(f_{\text{Eis}}) \in K[X], \quad \text{where } d := \deg(f_{\text{Eis}}).$$

After that we compute the field T for the polynomial f_{Eis}, employing the aforementioned algorithm "pReductionLFF", (see Algorithm 17.). Then, we follow the usual procedure for the splitting field computation. To be more precise, we factorize it until we find the field in which the polynomial f factors completely. In other words, if the initial polynomial f does not factor linearly, then we take the factor $\tau(f_{\text{Eis}})$ and we construct a field in which it factors completely. If the polynomial f is fully factored into linear factors, then the algorithm is terminated and the splitting field computation is over. If not, we repeat the same procedure for

$$\tau^2(f_{\text{Eis}}), \dots, \tau^{d-1}(f_{\text{Eis}})$$

until the splitting field of f is computed.

Remark 5.1.6

The above procedure is exactly the same with the one described in SplittingFieldLFF *algorithm (see Algorithm 9), for the splitting field computation. Specifically, the procedure can be described as follows: We choose the non-linear irreducible factors and we construct the tower of fields until the initial polynomial f splits into linear factors. The only difference with the process described in the mixed case is that in the latter case we construct the list with the non-linear factors from the beginning. After that, we choose factors included in this list and construct the tower of fields until the initial polynomial f factors completely.*

For the sake of clarity and completeness of the above case we give the following example.

Example 5.1.7
Let $f := X^4 + (t^2+t)X^2 + t^2X + t^2 \in K[X]$, where $K := \mathbb{F}_2((t))$. By factorization we have that $e = 2$, $\mathfrak{f} = 2$, $\pi = \alpha$ and $E := F(\beta)$, where β is a root of f_{Eis} and $F := K(\omega)$, where ω is a root of $g_1 := X^2 + X + 1$.

Let us now analyze the above steps. Firstly, we construct the unramified extension $U \cong \mathbb{F}_{2^2}((u_1)) \cong F$. We define the Frobenius automorphism

$$\tau : U \to U, \ \tau(\omega) = \omega^2 \text{ and } \tau(u_1) = u_1.$$

Then we consider f_{Eis} over U, and so $f_{\text{Eis}} = \tau(f_{\text{Eis}}) = X^2 + u_1X + u_1\omega^2$. After that, we can define the list

$$\text{NonLinFact} := [\tau^i(f_{\text{Eis}}),\ 0 \le i \le 2-1].$$

Indeed, we have that

$$\begin{aligned}\prod_{i=0}^{d-1}\tau^i(f_{\text{Eis}}) &= f_{\text{Eis}}\tau(f_{\text{Eis}})\\ &= (X^2+u_1X+u_1\omega^2)(X^2+u_1X+u_1\omega^4) = X^4+(u_1^2+u_1)X^2+u_1^2X+u_1^2 = f,\end{aligned}$$

as $t = u_1$. Thus, from the beginning we have determined the non-linear factors of f.

Then, we compute the field T of f_{Eis}, and we find out that $T \cong U$, that is $[T : U] = 1$.

$$\begin{tikzcd}
 & N := \mathrm{Spl}(f) \arrow[dl, no head] \arrow[dr, no head, "2"] & \\
E \arrow[d, no head, "e"'] & & K_1 \arrow[dll, no head, "2"'] \\
F \cong U \cong \mathbb{F}_{2^2}((u_1)) \cong T \arrow[d, no head, "\langle\tau\rangle"'] & & \\
K & &
\end{tikzcd}$$

Then, we construct the field extension generated by f_{Eis}. Let us denote it by $K_1 \cong \mathbb{F}_{2^2}((u_2))$. However, f does not factor into linear factors. Therefore, we construct the field extension generated by $\tau(f_{\text{Eis}})$, which is denoted by $N \cong \mathbb{F}_{2^2}((u_3))$, and thus f factors completely. Finally, $[N : K] = 8$.

For completeness we express the above procedure in an algorithmic form, which is given below.

Algorithm 20: Splitting Field: Mixed Case [SplittingFieldLFF_ANA_MC]

```
Input: A non-Eisenstein polynomial $f \in \mathcal{O}_K[X]$.
Output: The Splitting Field of $f$.
Construct the unramified extension $U/K$, where $U \cong \mathbb{F}_{q^{\mathfrak{f}}}((u_1))$, of degree $\mathfrak{f}$;
Consider $f_{\text{Eis}} \in U[X]$;
Define the Frobenius automorphism $\tau : U \to U,\ \tau(\omega) = \omega^p$ and $\tau(u_1) = u_1$;
Set NonLinFact $:= [\tau^i(f_{\text{Eis}}),\ 0 \le i \le d-1]$;
Compute the field $T$ for the polynomial $f_{\text{Eis}}$, that is $T \leftarrow$ pReductionLFF($f_{\text{Eis}}$);
repeat
    Construct field $K_i := \mathbb{F}_{q_i}((u_i))$ generted by $\tau^i(f_{\text{Eis}})$;
    Update the lists subst and data like in SplittingFieldLFF algorithm;
until $f$ factors completely;
Determine $\mathrm{Spl}(f) = K_l$, size $= [K_1 : K][K_2 : K_1]\cdots[K_l : K_{l-1}]$ and the roots rSF of $f$;
return $\mathrm{Spl}(f)$, size, rSF, subst, data;
```

5.1.1 The Splitting Field Algorithm

For the sake of completeness, we present an algorithm in which all the above cases are included. It is worth pointing out that this algorithm works provided that the initial polynomial is irreducible.

Before giving the algorithm, we establish some notation used throughout the algorithm. Firstly, $\mathrm{Spl}(f)$ stands for the splitting field of f over K and rSF denotes a list of the roots of f. Also, subst would denote a list with the substitutions that we make of f in order to compute the splitting field and data would be a dataset with the minimal polynomial in each intermediate field extension.

Algorithm 21: Splitting Field [SplittingFieldLFF_ANA_Plus]

```
Input: An irreducible polynomial $f \in \mathcal{O}_K[X]$.
Output: The splitting field of $f$.
if $f$ is Eisenstein then
    Call Algorithm 18, namely $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} \leftarrow$ SplittingFieldLFF_ANA($f$);
else
    Factorize $f$ and determime the inertia degree $\mathfrak{f}$, the ramification index $e$, the extension $E \cong K[X]\langle f\rangle$
      and its uniformizing element $\pi_E$;
    if $e = \deg(f)$ & $\mathfrak{f} = 1$: Totally Ramified Case then
        Call Algorithm 19, namely $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} :=$ SplittingFieldLFF_ANA_TR($f$);
    if $e = 1$ & $\mathfrak{f} = \deg(f)$: Totally Unramified Case then
        Call Algorithm 9, namely $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} :=$ SplittingFieldLFF($f$);
    if $e > 1$ & $\mathfrak{f} > 1$: Mixed Case then
        Call Algorithm 20, namely $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} :=$ SplittingFieldLFF_ANA_MC($f$);
return $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data}$;
```

Obviously, in the above algorithm we do not distinguish the case of tamely ramified extensions.

As we studied in the Section 1.9, devoted to the tamely ramified extensions, we can easily deal with such a case. To be more precise, as soon as we realize that the polynomial generates a tamely ramified extension, we can compute its splitting field without much effort by using the concepts and results of Section 1.9, and particularly Theorem 1.9.3.

Therefore, considering the above criterion, we can further develop our strategy for the splitting field computation of a given irreducible polynomial. This is encapsulated in the next algorithm.

Algorithm 22: Splitting Field [SplittingFieldLFF_final]

```
Input: An irreducible polynomial $f \in \mathcal{O}_k[X]$.
Output: The Splitting Field of $f$.
if $f$ generates a tamely ramified extension then
    Call the corresponding Algorithm 2, for tamely ramified extensions, namely
      $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} \leftarrow$ TameSplFldLFF($f$);
else
    Call Algorithm 21, namely $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data} :=$ SplittingFieldLFF_ANA_Plus($f$);
return $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data}$;
```

5.2 Efficiency of the Combined Approach

It is natural to ask about the efficiency of the splitting field algorithm developed in the so-called “Combined Approach” compared to the one analyzed in the previous approach “A Variation of the Basic Approach”, (see Section 3.2).

If the degree of a polynomial $f \in K[X]$, where $K := \mathbb{F}_q((t))$, is n, then the splitting field algorithm, described in section “A Variation of the Basic Approach”, (see Section 3.2), needs many factorizations in the worst case.

In addition, in this case the precision of the local function fields over which the factorization takes place increases gradually, which makes the factorization of the polynomials time-consuming.

In the case of an Eisenstein polynomial, Greve proved, in his PhD thesis [15], that Theorem 2.4.1 or Algorithm 6 gives us an important part of the splitting field with little effort. The fact that the field we are looking for is a p-extension over the field T greatly restricts the factoring behavior of f. Over T, or extensions of T, only factors of p-power degree can occur. As a result, the number of factorizations in the worst case is reduced. Thus, a combination of Algorithm 6 (p-Reduction Algorithm), especially in our case Algorithm 17, and Algorithm 9 (SplittingFieldLFF) over T would be a first, improved approach to the computation of the splitting field. It should be noted that for many examples this yields a big step towards the direction of the splitting field. In particular, when we get a constant field extension in this way, we save time in the factorization algorithm, where the mixed case is more difficult than the case where the polynomial is completely ramified.

Moreover, since T is totally tamely ramified extension over its constant field, the uniformizing element can be computed easily. The reason is that the latter extension is a Kummer extension, and therefore we can choose the uniformizing element

$$\pi_T = u_2 = \sqrt[e_0]{\omega t},$$

where t is the uniformizing element of K and ω is an element of the constant field extension. This simplifies the computation of the corresponded isomorphisms Φ_i described in Chapter 3, and hence leads to simpler represented embedded elements

$$\Phi_1(t),\ \Phi_{j+1}(u_j),\quad \text{with}\ \ 1 \leq j \leq \ell - 1.$$

After computing T, we proceed with the usual splitting field approach. Nevertheless, we emphasize that the factorization of $f \in T[X]$ over the field T results in simpler represented irreducible factors, which in turn contributes to a better represented tower for the construction of the splitting field.

Furthermore, the computation of the field T affects the behavior of the precision. In fact, it can influence on a better raise in the precision needed at each intermediate step of the tower.

Let us now compare the two approaches -"A Variation of the Basic Approach" and "Combined Approach"- of splitting field computation in practice. To do so, we provide some relevant examples.

Example 5.2.1
Let $f(X) = X^8 + t^9X^7 + t^{15}X^6 + tX^4 + t \in k[X]$, *where* $k := \mathbb{F}_2((t))$. *The polynomial* f *is an Eisenstein polynomial and its ramification polygon consists of two segments with absolute value of slopes* $[67/3, 1]$. *Having computed its splitting field we know that*

$$\mathrm{Spl}(f/k) = \mathbb{F}_{2^2}((u_4)) \ \ and\ \ |\mathrm{Spl}(f/k)| = 192 = 2^6 \cdot 3.$$

Also, we get access to its roots, and so we can determine the u_4*-valuation of its roots, which is given below:*

$$v_{u_4}(\alpha_i - \alpha) := \{v_{u_4}(\alpha_i - \alpha)\ :\ 1 \leq i \leq 8\} = \{24^4, 280^3, \infty\}.$$

$$\begin{tikzcd}
& \tilde{k}_3[X]/\langle f_2\rangle \cong \ \mathbb{F}_{2^2}((\tilde{u}_4)) =: \tilde{k}_4 \cong N \cong k_2 := \mathbb{F}_{2^2}((u_4)) \cong k_1[X]/\langle f_1\rangle \arrow[dl, dash, "4"'] \arrow[dr, dash, "4"] & \\
\tilde{k}_2[X]/\langle f_{1,1}\rangle \cong \tilde{k}_3 := \mathbb{F}_{2^2}((\tilde{u}_3)) \arrow[d, dash, "2"'] & & k_1 := \mathbb{F}_{2^2}((u_3)) \ \cong T[X]/\langle f\rangle \arrow[dd, dash, "8"] \\
\tilde{k}_1[X]/\langle f_1\rangle \cong \tilde{k}_2 := \mathbb{F}_2((\tilde{u}_2)) \arrow[d, dash, "3"'] & & \\
k[X]/\langle f\rangle \cong \tilde{k}_1 := \mathbb{F}_2((\tilde{u}_1)) \arrow[dr, dash, "8"'] & & T := \mathbb{F}_{2^2}((u_2)) \quad f \in T[X] \arrow[dl, dash, "6"] \\
& k := \mathbb{F}_2((t)) \quad f \in k[X] &
\end{tikzcd}$$

Computing $\mathrm{Spl}(f)$ *by using the so-called "A Variation of the Basic Approach", the time needed for its computation is* ***2.59 sec*** *and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(\tilde{u}_4) = 2798$. *On the other hand, we employ the splitting field algorithm based on the so-called "Combined Approach" for its determination. In that case, though, the time needed for its computation is* ***1.79 sec*** *and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(u_4) = 3021$.

We begin with computing $\mathrm{Spl}(f)$ *by using the so-called "A Variation of the Basic Approach". Let us give its description step by step.*
Step 1: Time: 0.17 sec
We take the polynomial $f \in k[X]$, *which is Eisenstein, to generate the extension* $\tilde{k}_1/k$. *Thus, the field* $\tilde{k}_1 := \mathbb{F}_2((\tilde{u}_1))$ *with* $\mathrm{prec}(\tilde{u}_1) = 2083$. *Now we factorize* f *and get*

$$f = (X - \alpha_1)f_1f_2 \in \tilde{k}_1[X],$$

with $\deg(f_1) = 3$ *and* $\deg(f_2) = 4$. *We store the root of* f *and proceed to the next step.*
Step 2: Time: 0.63 sec

We choose f_1 to construct the extension $\tilde{k}_2/\tilde{k}_1$. Thus, the field $\tilde{k}_2 := \mathbb{F}_2((u_2))$ with $\operatorname{prec}(\tilde{u}_2) = 2178$. *We consider f_1 over $\tilde{k}_2$ and by factorizing it over $\tilde{k}_2$ we get that*

$$f_1 = (X - \alpha_2) f_{1,1} \in \tilde{k}_2[X],$$

with $\deg(f_{1,1}) = 2$. *We upgrade the stored factors of f so as to be over $\tilde{k}_2[X]$. Then, we factorize $f_2 \in \tilde{k}_2[X]$ and get that it is irreducible. So, we store the irreducible factors as well as the roots of f and proceed to the next step.*

Step 3: Time:* 0.08 *sec

We choose $f_{1,1}$ to construct the extension $\tilde{k}_3/\tilde{k}_2$. At this step, we get a non-trivial inertia degree, so we have a constant field extension. Thus, the field $\tilde{k}_3 := \mathbb{F}_{2^2}((u_3))$ with $\operatorname{prec}(\tilde{u}_3) = 2178$. *We consider $f_{1,1}$ over $\tilde{k}_3$ and by factorizing it over $\tilde{k}_3$ we get that*

$$f_{1,1} = (X - \alpha_3)(X - \alpha_4) \in \tilde{k}_3[X].$$

We upgrade the stored factors of f to be over $\tilde{k}_3[X]$. We factorize $f_2 \in \tilde{k}_3[X]$ and get that it is irreducible. We store the irreducible factors as well as the roots of f and proceed to the next step.

Step 4: Time:* 1.63 *sec

We choose f_2 to construct the extension $\tilde{k}_4/\tilde{k}_3$. Thus, the field $\tilde{k}_4 := \mathbb{F}_{2^2}((u_4))$ with $\operatorname{prec}(\tilde{u}_4) = 2798$. *We consider f_2 over $\tilde{k}_4$ and by factorizing it over $\tilde{k}_4$ we get that*

$$f_2 = (X - \alpha_5)(X - \alpha_6)(X - \alpha_7)(X - \alpha_8) \in \tilde{k}_4[X].$$

Therefore, the polynomial f splits into linear factors, which completes its splitting field computation.

Let us now describe the splitting field algorithm based on the so-called "Combined Approach" in detail. Given $f \in k[X]$ we compute the field T which is $T := \mathbb{F}_{2^2}((u_2))$ with respect to the the relation $\Phi_2(\Phi_1(t)) = u_2^3$ of degree $[T : k] = 6$ and $\operatorname{prec}(u_2) = 914$, *and then we consider f over T as*

$$\operatorname{Spl}(f/k) = \operatorname{Spl}(f/T).$$

By factorizing it over T, we get that f is irreducible.

Step 1: Time:* 0.14 *sec

We take the polynomial $f \in T[X]$, which is irreducible but not Eisenstein, to generate the extension k_1/T. Thus, the field $k_1 := \mathbb{F}_{2^2}((u_3))$ with $\operatorname{prec}(u_3) = 2701$. *Now we factorize f and get*

$$f = \prod_{i=1}^{4} (X - \alpha_i) f_1 \in k_1[X],$$

with $\deg(f_1) = 4$. *We store the roots of f.*

Step 2: Time:* 1.39 *sec

We choose f_1 to construct the extension k_2/k_1. Thus, the field $k_2 := \mathbb{F}_{2^2}((u_4))$ with $\operatorname{prec}(u_4) = 3021$. *We consider f_1 over k_2 and by factorizing it over k_2 we get that*

$$f_1 = \prod_{i=4}^{8} (X - \alpha_i) \in k_2[X].$$

The polynomial f factors completely, which completes its splitting field computation.

Example 5.2.2

Let $f(X) = X^{12} + t^9 X^5 + t \in k[X]$, where $k := \mathbb{F}_3((t))$. The polynomial f is an Eisenstein polynomial and its ramification polygon consists of two segments with absolute value of its slopes $[101/2, 0]$. Having computed its splitting field we know that

$$\operatorname{Spl}(f/k) = \mathbb{F}_{3^2}((u_4)) \ \text{ and } \ |\operatorname{Spl}(f/k)| = 144 = 2^4 \cdot 3^2.$$

Also, we get access to its roots, and so we can determine the u_4-valuation of its roots, which is given below:

$$v_{u_4}(\alpha_i - \alpha) := \{v_{u_4}(\alpha_i - \alpha) \ : \ 1 \le i \le 8\} = \{309^2, 6^9 \infty\}.$$

$$\begin{tikzcd}
 & \tilde{k}_3[X]/\langle f_{1,1}\rangle \cong \mathbb{F}_{3^2}((\tilde{u}_4)) =: \tilde{k}_4 \cong N \cong k_2 := \mathbb{F}_{3^2}((u_4)) \cong k_1[X]/\langle f_2\rangle \arrow[dl, dash, "2"'] \arrow[dr, dash, "3"] & \\
\tilde{k}_2[X]/\langle f_1\rangle \cong \tilde{k}_3 := \mathbb{F}_{3^2}((\tilde{u}_3)) \arrow[d, dash, "3"'] & & k_1 := \mathbb{F}_{3^2}((u_3)) \cong T[X]/\langle f_1\rangle \arrow[dd, dash, "3"] \\
\tilde{k}_2 := \mathbb{F}_{3^2}((\tilde{u}_2)) \arrow[d, dash, "2"'] & & \\
k[X]/\langle f\rangle \cong \tilde{k}_1 := \mathbb{F}_3((\tilde{u}_1)) \arrow[dr, dash, "12"'] & & T := \mathbb{F}_{3^2}((u_2)) \quad f \in T[X] \arrow[dl, dash, "16"] \\
 & k := \mathbb{F}_3((t)) \quad f \in k[X] &
\end{tikzcd}$$

Computing $\mathrm{Spl}(f)$ *by using the so-called "A Variation of the Basic Approach", the time needed for its computation is* **63.21** *sec and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(\tilde{u}_4) = 9414$. *On the other hand, we employ the splitting field algorithm based on the so-called "Combined Approach" for its determination. In that case, though, the time needed for its computation is* **4.83** *sec and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(u_4) = 4409$.

We begin with computing $\mathrm{Spl}(f)$ *by using the so-called "A Variation of the Basic Approach". Let us give its description step by step.*

Step 1: Time: **1.72** ***sec***

We take the polynomial $f \in k[X]$, *which is Eisenstein, to generate the extension* $\tilde{k}_1/k$. *Thus, the field* $\tilde{k}_1 := \mathbb{F}_3((\tilde{u}_1))$ *with* $\mathrm{prec}(\tilde{u}_1) = 8897$. *Now we factorize* f *and get*

$$f = (X - \alpha_1) f_1 f_2 f_3 \in \tilde{k}_1[X],$$

with $\deg(f_1) = 3$, $\deg(f_2) = 2$, *and* $\deg(f_3) = 6$. *We store the root of* f *and proceed to the next step.*

Step 2: Time: **2.99** ***sec***

At this step, the inertia degree of the irreducible factors is 2, *so we have a constant field extension. Thus, the field* $\tilde{k}_2 := \mathbb{F}_{3^2}((u_2))$ *with* $\mathrm{prec}(\tilde{u}_2) = 8897$. *We upgrade the stored factors of* f *so as to be over* $\tilde{k}_2[X]$. *Then, we factorize* $f_1, f_2, f_3 \in \tilde{k}_2[X]$ *and get that they are still irreducible. So, we store them as well as the roots of* f *and proceed to the next step.*

Step 3: Time: **36.28** ***sec***

We choose f_1 *to construct the extension* $\tilde{k}_3/\tilde{k}_2$. *Thus, the field* $\tilde{k}_3 := \mathbb{F}_{2^2}((u_3))$ *with* $\mathrm{prec}(\tilde{u}_3) = 9076$. *We consider* f_1 *over* $\tilde{k}_3$ *and by factorizing it over* $\tilde{k}_3$ *we get that*

$$f_{1,1} = (X - \alpha_2) f_{1,1} \in \tilde{k}_3[X],$$

with $\deg(f_{1,1}) = 2$. *We upgrade the stored factors of* f *so as to be over* $\tilde{k}_3[X]$. *Then, we factorize the stored factors. It holds that* $f_2 \in \tilde{k}_3[X]$ *splits, and thus*

$$f_2 = (X - \alpha_3)(X - \alpha_4) \in \tilde{k}_3[X].$$

The factor $f_3 \in \tilde{k}_3[X]$ *is still irreducible. So, we store the irreducible factors as well as the roots of* f *and proceed to the next step.*

Step 4: Time: **20.04** ***sec***

We choose $f_{1,1}$ *to construct the extension* $\tilde{k}_4/\tilde{k}_3$. *Thus, the field* $\tilde{k}_4 := \mathbb{F}_{3^2}((u_4))$ *with* $\mathrm{prec}(\tilde{u}_4) = 9414$. *We consider* $f_{1,1}$ *over* $\tilde{k}_4$ *and by factorizing it over* $\tilde{k}_4$ *we get that*

$$f_{1,1} = (X - \alpha_5)(X - \alpha_6) \in \tilde{k}_4[X].$$

We upgrade the stored factors of f *so as to be over* $\tilde{k}_4[X]$. *It holds that* $f_3 \in \tilde{k}_4[X]$ *splits, and thus*

$$f_3 = \prod_{i=7}^{12} (X - \alpha_i) \in \tilde{k}_4[X].$$

Therefore, the polynomial f *splits into linear factors, which completes its splitting field computation.*

Let us now describe the splitting field algorithm based on the so-called "Combined Approach" in detail. Given $f \in k[X]$ *we compute the field* T *which is* $T := \mathbb{F}_{3^2}((u_2))$ *with respect to the the relation* $\Phi_2(\Phi_1(t)) = 2u_2^8$ *of degree* $[T : k] = 16 = (2) \cdot (2^3)$ *and* $\mathrm{prec}(u_2) = 2281$, *and then we consider* f *over* T *as* $\mathrm{Spl}(f/k) = \mathrm{Spl}(f/T)$. *By factorizing it over* T, *we get that*

$$f = f_1 f_2 f_3 f_4 \in T[X], \quad \textit{with } \deg(f_i) = 3, \quad \textit{for } 1 \le i \le 4.$$

Step 1: Time: **1.73** ***sec***

We take the polynomial $f_1 \in T[X]$ *to generate the extension* k_1/T. *Thus, the field* $k_1 := \mathbb{F}_{3^2}((u_3))$ *with* $\mathrm{prec}(u_3) = 4365$. *Now we factorize* f_1 *and get*

$$f_1 = \prod_{i=1}^{3} (X - \alpha_i) \in k_1[X].$$

We upgrade the stored factors of f *to be over* $k_1[X]$. *Then, we factorize* $f_2, f_3, f_4 \in k_1[X]$ *and get that they are still irreducible. We store them as well as the roots of* f *and proceed to the next step.*

Step 2: Time: **2.85** ***sec***

We choose f_2 *to construct the extension* k_2/k_1. *Thus, the field* $k_2 := \mathbb{F}_{3^2}((u_4))$ *with* $\mathrm{prec}(u_4) = 4409$. *We consider* f_2 *over* k_2 *and by factorizing it over* k_2 *we get that*

$$f_2 = \prod_{i=4}^{6} (X - \alpha_i) \in k_2[X].$$

We upgrade the stored factors of f to be over $\tilde{k}_2[X]$. It holds that $f_3, f_4 \in k_2[X]$ split, and thus

$$f_3 = \prod_{i=7}^{9}(X-\alpha_i) \in k_2[X] \ \ \textit{and } f_4 = \prod_{i=10}^{12}(X-\alpha_i) \in k_2[X].$$

The polynomial f factors completely, which completes its splitting field computation.

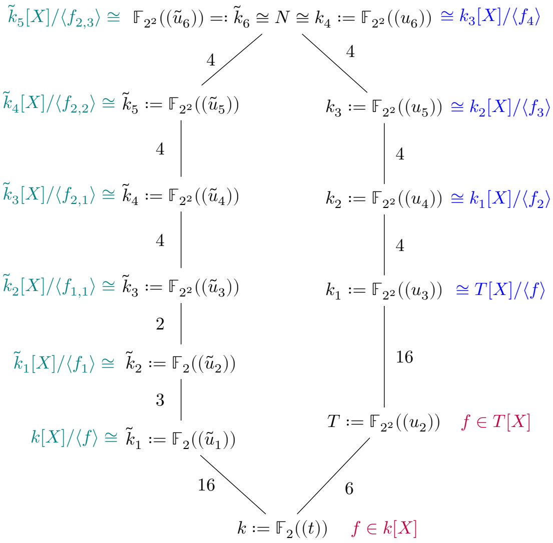
Example 5.2.3

Let $f(X) = X^{16} + t^5X^{10} + t^7X^9 + t^2X^8 + tX^4 + t \in k[X]$, where $k := \mathbb{F}_2((t))$. The polynomial f is an Eisenstein polynomial and its ramification polygon consists of two segments with absolute value of its slopes $[101/3, 1/3]$. Having computed its splitting field we know that

$$\mathrm{Spl}(f/k) = \mathbb{F}_{2^2}((u_6)) \ \textit{ and } \ |\mathrm{Spl}(f/k)| = 6144 = 2^{11}\cdot 3.$$

Also, we get access to its roots, and so we can determine the u_4-valuation of its roots, which is given below:

$$v_{u_6}(\alpha_i - \alpha) := \{v_{u_6}(\alpha_i - \alpha) \ : \ 1 \le i \le 16\} = \{256^{12}, 6656^3, \infty\}.$$


$\tilde{k}_5[X]/\langle f_{2,3}\rangle \cong \mathbb{F}_{2^2}((\tilde{u}_6)) =: \tilde{k}_6 \cong N \cong k_4 := \mathbb{F}_{2^2}((u_6)) \cong k_3[X]/\langle f_4\rangle$
4
4
$\tilde{k}_4[X]/\langle f_{2,2}\rangle \cong \tilde{k}_5 := \mathbb{F}_{2^2}((\tilde{u}_5))$
$k_3 := \mathbb{F}_{2^2}((u_5)) \cong k_2[X]/\langle f_3\rangle$
4
4
$\tilde{k}_3[X]/\langle f_{2,1}\rangle \cong \tilde{k}_4 := \mathbb{F}_{2^2}((\tilde{u}_4))$
$k_2 := \mathbb{F}_{2^2}((u_4)) \cong k_1[X]/\langle f_2\rangle$
4
4
$\tilde{k}_2[X]/\langle f_{1,1}\rangle \cong \tilde{k}_3 := \mathbb{F}_{2^2}((\tilde{u}_3))$
$k_1 := \mathbb{F}_{2^2}((u_3)) \cong T[X]/\langle f\rangle$
2
$\tilde{k}_1[X]/\langle f_1\rangle \cong \tilde{k}_2 := \mathbb{F}_2((\tilde{u}_2))$
16
3
$T := \mathbb{F}_{2^2}((u_2))$ $f \in T[X]$
$k[X]/\langle f\rangle \cong \tilde{k}_1 := \mathbb{F}_2((\tilde{u}_1))$
16
6
$k := \mathbb{F}_2((t))$ $f \in k[X]$

Computing $\mathrm{Spl}(f)$ *by using the so-called "A Variation of the Basic Approach", the time needed for its computation is* ***24145.43 sec*** $\sim$ ***6.7h*** *and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(\tilde{u}_6) = 57620$. *On the other hand, we employ the splitting field algorithm based on the so-called "Combined Approach" for its determination. In that case, though, the time needed for its computation is* ***8227.4 sec*** $\sim$ ***2.2h*** *and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(u_6) = 43988$.

We begin with computing $\mathrm{Spl}(f)$ *by using the so-called "A Variation of the Basic Approach". Let us give its description step by step.*

Step 1: Time: 774.05 sec $\sim$ ***12.9min***

We take the polynomial $f \in k[X]$, which is Eisenstein, to generate the extension $\tilde{k}_1/k$. Thus, the field $\tilde{k}_1 := \mathbb{F}_2((\tilde{u}_1))$ with $\mathrm{prec}(\tilde{u}_1) = 8027$. *Now we factorize f and get*

$$f = (X-\alpha_1)f_1f_2 \in \tilde{k}_1[X],$$

with $\deg(f_1) = 3$ *and* $\deg(f_3) = 12$. *We store the root of f and proceed to the next step.*

Step 2: Time: 653.48 sec $\sim$ ***10.9min***

We choose f_1 to construct the extension $\tilde{k}_2/\tilde{k}_1$. Thus, the field $\tilde{k}_2 := \mathbb{F}_2((u_2))$ with $\mathrm{prec}(\tilde{u}_2) = 8155$. *We consider f_1 over $\tilde{k}_2$ and by factorizing it over $\tilde{k}_2$ we get that*

$$f_1 = (X-\alpha_2)f_{1,1} \in \tilde{k}_3[X],$$

with $\deg(f_{1,1}) = 2$. *We upgrade the stored factors of f to be over $\tilde{k}_2[X]$. We factorize $f_2 \in \tilde{k}_2[X]$ and get it is still irreducible. We store them as well as the roots of f and proceed to the next step.*

Step 3: Time: 246.19 sec $\sim$ ***4.1min***

We choose $f_{1,1}$ to construct the extension $\tilde{k}_3/\tilde{k}_2$. At this step, we get a non-trivial inertia degree, so we have a constant field extension. Thus, the field $\tilde{k}_3 := \mathbb{F}_{2^2}((u_3))$ with $\mathrm{prec}(\tilde{u}_3) = 8155$. *We consider $f_{1,1}$ over $\tilde{k}_3$ and by factorizing it over $\tilde{k}_3$ we get that*

$$f_{1,1} = (X-\alpha_3)(X-\alpha_4) \in \tilde{k}_3[X].$$

We upgrade the stored factors of f so as to be over $\tilde{k}_3[X]$. Then, we factorize the stored factors. It holds that $f_2 \in \tilde{k}_3[X]$ factors, and thus

$$f_2 = f_{2,1}f_{2,2}f_{2,3} \in \tilde{k}_3[X], \quad \textit{with } \deg(f_{2,i}) = 4 \quad \textit{with } 1 \leq i \leq 3.$$

So, we store the irreducible factors as well as the roots of f and proceed to the next step.

Step 4: Time:* 604.14 *sec* $\sim$ 10*min

We choose $f_{2,1}$ to construct the extension $\tilde{k}_4/\tilde{k}_3$. Thus, the field $\tilde{k}_4 := \mathbb{F}_{2^2}((u_4))$ with $\operatorname{prec}(\tilde{u}_4) =$ 10944. *We consider $f_{2,1}$ over $\tilde{k}_4$ and by factorizing it over $\tilde{k}_4$ we get that*

$$f_{2,1} = \prod_{i=5}^{8}(X - \alpha_i) \in \tilde{k}_4[X].$$

We upgrade the stored factors of f so as to be over $\tilde{k}_4[X]$. Then, we factorize $f_{2,2}, f_{2,3} \in \tilde{k}_4[X]$, and get they are still irreducible. So, we store the irreducible factors as well as the roots of f and proceed to the next step.

Step 5: Time:* 8556.89 *sec* $\sim$ 2.3*h

We choose $f_{2,2}$ to construct the extension $\tilde{k}_5/\tilde{k}_4$. Thus, the field $\tilde{k}_5 := \mathbb{F}_{2^2}((u_5))$ with $\operatorname{prec}(\tilde{u}_5) =$ 43776. *We consider $f_{2,2}$ over $\tilde{k}_5$ and by factorizing it over $\tilde{k}_5$ we get that*

$$f_{2,2} = \prod_{i=9}^{12}(X - \alpha_i) \in \tilde{k}_5[X].$$

We upgrade the stored factors of f to be over $\tilde{k}_5[X]$. We factorize $f_{2,3} \in \tilde{k}_5[X]$, and get it is still irreducible. We store the irreducible factors as well as the roots of f and proceed to the next step.

Step 6: Time:* 12603.70 *sec* $\sim$ 3.5*h

We choose $f_{2,3}$ to construct the extension $\tilde{k}_6/\tilde{k}_5$. Thus, the field $\tilde{k}_6 := \mathbb{F}_{2^2}((u_6))$ with $\operatorname{prec}(\tilde{u}_6) =$ 57620. *We consider $f_{2,3}$ over $\tilde{k}_6$ and by factorizing it over $\tilde{k}_6$ we get that*

$$f_{2,3} = \prod_{i=13}^{16}(X - \alpha_i) \in \tilde{k}_6[X].$$

Therefore, the polynomial f splits into linear factors, which completes its splitting field computation.

Let us describe the splitting field algorithm based on the so-called "Combined Approach" in detail. Given $f \in k[X]$ we compute the field T which is $T := \mathbb{F}_{2^2}((u_2))$ with respect to the the relation $\Phi_2(\Phi_1(t)) = u_2^3$ of degree $[T : k] = 6 = (2) \cdot (3)$ and $\operatorname{prec}(u_2) = 948$, *and then we consider f over T as $\operatorname{Spl}(f/k) = \operatorname{Spl}(f/T)$. By factorizing it over T, we get that f is irreducible.*

Step 1: Time:* 21.21 *sec

We take the polynomial $f \in T[X]$ to generate the extension k_1/T. Thus, the field $k_1 := \mathbb{F}_{2^2}((u_3))$ with $\operatorname{prec}(u_3) = 2704$. *Now we factorize f and get*

$$f = \prod_{i=1}^{4}(X - \alpha_i)f_1 f_2 f_3 \in k_1[X], \ \textit{with } \deg(f_j) = 4 \ \forall j = 1, 2, 3.$$

We upgrade the stored factors of f so as to be over $k_1[X]$. So, we store the irreducible factors as well as the roots of f and proceed to the next step.

Step 2: Time:* 492.04 *sec* $\sim$ 8.2*min

We choose f_1 to construct the extension k_2/k_1. Thus, the field $k_2 := \mathbb{F}_{2^2}((u_4))$ with $\operatorname{prec}(u_4) =$ 10944. *We consider f_1 over k_2 and by factorizing it over k_2 we get that*

$$f_1 = \prod_{i=5}^{8}(X - \alpha_i) \in k_2[X].$$

We upgrade the stored factors of f so as to be over $k_2[X]$. It holds that $f_2, f_3 \in k_2[X]$ are still irreducible. So, we store them as well as the roots of f and proceed to the next step.

Step 3: Time:* 3340.16 *sec* $\sim$ 55.6*min

We choose f_2 to construct the extension k_3/k_2. Thus, the field $k_3 := \mathbb{F}_{2^2}((u_5))$ with $\operatorname{prec}(u_5) =$ 43776. *We consider f_2 over k_3 and by factorizing it over k_3 we get that*

$$f_2 = \prod_{i=9}^{12}(X - \alpha_i) \in k_3[X].$$

We upgrade the stored factors of f so as to be over $k_3[X]$. It holds that $f_3 \in k_3[X]$ is still irreducible. So, we store it as well as the roots of f and proceed to the next step.

Step 4: Time:* 3888.85 *sec* $\sim$ 1*h
We choose f_3 to construct the extension k_4/k_3. Thus, the field $k_4 := \mathbb{F}_{2^2}((u_6))$ with $\operatorname{prec}(u_6) =$ 43988. *We consider f_3 over k_4 and by factorizing it over k_4 we get that*

$$f_3 = \prod_{i=13}^{16} (X - \alpha_i) \in k_4[X].$$

The polynomial f factors completely, which completes its splitting field computation.

5.3 Examples: Comparison of Version 1 to Version 2

For the sake of completeness we list a table of examples in which we compare the two approaches -“A Variation of the Basic Approach”, (Ver1), and “Combined Approach”, (Ver2),- for the splitting field computation. This table can be found in Section A.1.

5.4 Improvement of Splitting Field Computations

One of the drawbacks of using our method by converting every intermediate extension field of $K := \mathbb{F}_q((t))$ to a field of the form $\mathbb{F}_{q_i}((u_i))$ is that we have to increase the precision and find the embedded elements u_i. Having determined the embedded elements u_i, we use them to update all the irreducible factors of f stored up to that point. As a result, the remaining polynomials are getting more and more complicated as the valuation of their coefficients becomes increasingly large, which in turn leads to time-consuming factorization of the polynomials.

With the intention of improving the above behavior, we want to make the candidate factor which will be used to generate the current extension “nicer”. This yields to speed up the factorization, and the whole process in general.

To do so, we will follow the idea that in every intermediate step we construct the extension by using the corresponding Eisenstein polynomial g_i of the current candidate factor f_i such that

$$K_{i-1}[X]/\langle f_i \rangle \cong K_{i-1}[X]/\langle g_i \rangle.$$

As we have already described, those Eisenstein polynomials are the minimal polynomials of the computed uniformizing elements, and those can be computed by using the optional parameter “Extensions” in the factorization algorithm provided by Magma. Certainly, the irreducible factor and the corresponding Eisenstein polynomial generate the same field extension, since

$$\operatorname{Spl}(f_i/K_{i-1}) = \operatorname{Spl}(g_i/K_{i-1}).$$

Therefore, we can use the much nicer Eisenstein polynomial for the rest of the splitting field computation. The mathematical realization of this idea is easy, but the implementation needs a lot of attention.

Let $f \in K[X]$ be an Eisenstein polynomial and $K_1 = K[X]/\langle f \rangle$. By factorizing f, we get

$$f = \prod_i (X - \alpha_i) \prod_{j=1}^{r} f_j \in K_1[X], \quad \text{and so } \operatorname{Spl}(f) = \operatorname{Spl}(f_1 \cdots f_r).$$

Then for every $1 \le j \le r$ we compute $g_i \in K_1[X]$ such that g_i is Eisenstein and

$$K_1[X]/\langle f_j \rangle \cong K_1[X]/\langle g_j \rangle.$$

Remark 5.4.1
Under the assumptions and conditions stated above, it holds that

$$\operatorname{Spl}(f_1 \cdots f_r/K_1) = \operatorname{Spl}(g_1 \cdots g_r/K_1).$$

Proof. Note that $\operatorname{Spl}(f_1 \cdots f_r/K_1)$ is the composite $\operatorname{Spl}(f_1/K_1) \cdots \operatorname{Spl}(f_r/K_1)$, and similarly we get $\operatorname{Spl}(g_1 \cdots g_r/K_1) = \operatorname{Spl}(g_1/K_1) \cdots \operatorname{Spl}(g_r/K_1)$. Since for every $1 \le j \le r$ it holds that $\operatorname{Spl}(f_j/K_1) = \operatorname{Spl}(g_j/K_1)$, the remark follows. □

There are many advantages of applying this approach. The main one of them is that the valuations of the coefficients of $g_i \in K_{i-1}[X]$ are smaller than the one of f_i. Therefore, working with those polynomials results in the need for less precision throughout the computations, thereby speeding up the factorization of not only the current candidate generating polynomial but also the stored irreducible factors. Moreover, this approach leads to simplify the check of the stored irreducible factors. Particularly, by checking their corresponding Eisenstein polynomials in each step results in keeping as simple as possible the stored irreducible factors. By this we mean that the stored factors have coefficients with small valuation compared with that of the original factors of the initial polynomial. As a result, checking the irreducibility of those polynomials requires less precision in order to complete successfully. In addition, the representation of the stored irreducible factors is not as complicated as that of the initial factors would be. Consequently, the arithmetic using those polynomials is faster.

Nevertheless, applying this approach, we should bear in mind that we have to make appropriate adjustments so as to obtain the roots of the initial polynomial $f \in K[X]$. Precisely, in every step we store the corresponding formula $\pi_{K_i} = A(\alpha_i)$, (see (5.1)), as analyzed in Section 3.2, that comes along with the extension for each irreducible factor. Particularly, this formula gives a relation between the roots of f_i and g_i, (see Propositions 5.1.3 and 5.1.4). In order to get the roots of f, we recursively use that relation in every step.

The aforementioned approach can be described in the following diagram.

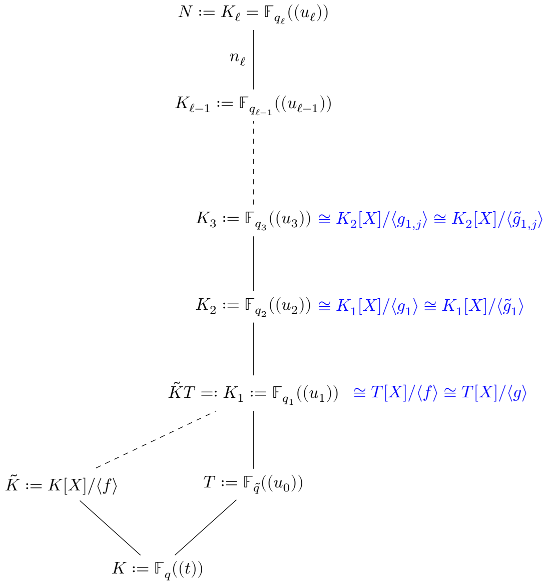

$N := K_\ell = \mathbb{F}_{q_\ell}((u_\ell))$
n_ℓ
$K_{\ell-1} := \mathbb{F}_{q_{\ell-1}}((u_{\ell-1}))$
$K_3 := \mathbb{F}_{q_3}((u_3)) \cong K_2[X]/\langle g_{1,j}\rangle \cong K_2[X]/\langle \tilde{g}_{1,j}\rangle$
$K_2 := \mathbb{F}_{q_2}((u_2)) \cong K_1[X]/\langle g_1\rangle \cong K_1[X]/\langle \tilde{g}_1\rangle$
$\tilde{K}T =: K_1 := \mathbb{F}_{q_1}((u_1)) \cong T[X]/\langle f\rangle \cong T[X]/\langle g\rangle$
$\tilde{K} := K[X]/\langle f\rangle$
$T := \mathbb{F}_{\tilde{q}}((u_0))$
$K := \mathbb{F}_q((t))$

Figure 5.3: **Splitting Field: Improved Approach**

Given a polynomial $f \in K[X]$ we compute the field T, and then we consider f over T as

$$\mathrm{Spl}(f/K) = \mathrm{Spl}(f/T).$$

We check if $f \in T[X]$ is Eisenstein and if so then we choose it as the defining polynomial of the extension K_1/T. Otherwise, we take the corresponding Eisenstein polynomial g of f and generate the extension K_1/T as $\mathrm{Spl}(f/T) = \mathrm{Spl}(g/T)$. At this point we use the formula $\pi_0 = \pi_T \in T[X]$, (see (5.1)), which gives the relation between the roots of g and f in order to compute the formula $B_0(X) \in T[X]$, (see (5.2)). For more details see Propositions 5.1.3 and 5.1.4, and in general the paragraph "Totally Ramified Case". So we store them as follows $\langle g, B_0\rangle$, and proceed to the next step. Now we factorize g and get

$$g = \prod_i (X - \tilde{\alpha}_i) \prod_j g_j \in K_1[X].$$

We compute the roots of f which are of the form

$$\alpha_i = B_0(\tilde{\alpha}_i) \text{ for every } i$$

and store them. Also, for every factor g_j we determine its corresponding Eisenstein polynomial $\tilde{g}_j$ and the formula $B_{1,j}$ by the relation $\pi_{1,j}$ of their roots, and then we store the factors

$$\langle \tilde{g}_j, B_0(B_{1,j}(X))\rangle \text{ for every } j.$$

We proceed to the next step and compute $\mathfrak{f}(K_1/T) = \operatorname{lcm}(\mathfrak{f}(\tilde{g}_j)\ ;\ \forall j)$, where $\mathfrak{f}(\tilde{g}_j)$ stands for the inertia degree of $\tilde{g}_j$, (see Remark 3.2.8 and Algorithm 9, Step 11). If $\mathfrak{f}(K_1/T) > 1$, then we construct the constant field extension and proceed to the next step. Otherwise, we choose $\tilde{g}_1$ to construct the extension K_2/K_1, as

$$\operatorname{Spl}(g_1/K_1) = \operatorname{Spl}(\tilde{g}_1/K_1).$$

By factorizing it over K_2, we get that

$$\tilde{g}_1 = \prod_i (X - \tilde{\alpha}_i) \prod_l \tilde{g}_{1,l} \in K_2[X].$$

We determine the roots of f which are

$$\alpha_i = B_0(B_{1,1}(\tilde{\alpha}_i)) \text{ for every } i$$

and store them. Also, for every factor $\tilde{g}_{1,l}$ we determine its corresponding Eisenstein polynomial $\tilde{\tilde{g}}_{1,l}$ and the formula $B_{2,1,l}$ by the relation $\pi_{2,1,l}$ of their roots, and then we store the factors

$$\langle \tilde{\tilde{g}}_{1,l}, B_0(B_{1,1}(B_{2,1,l}(X)))\rangle \text{ for every } l.$$

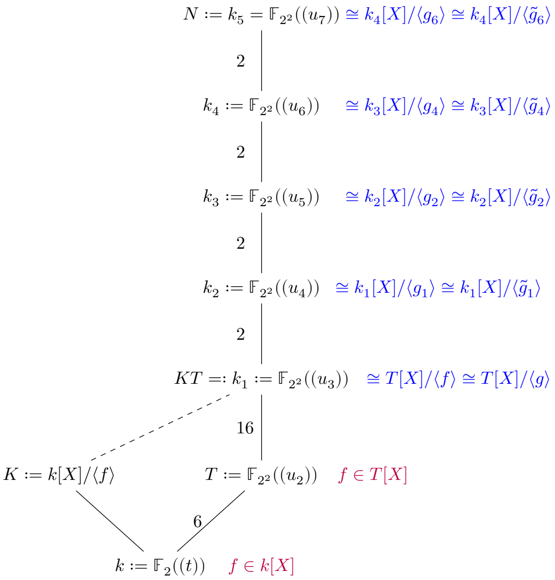
In addition to them, we factorize and upgrade the old factors which are of the following form

$$\langle \tilde{\tilde{g}}_j, B_0(B_{1,j}(B_{2,j}(X)))\rangle \text{ for every } j.$$

Similarly, we repeat the above process until f factors completely.

For the sake of clarity and completeness of the above approach we give the next examples.

Example 5.4.2
*Let $f(X) = X^{16} + tX^{15} + tX^6 + tX^4 + t \in k[X]$, where $k := \mathbb{F}_2((t))$. Then, $\operatorname{Spl}(f/k) = \mathbb{F}_{2^2}((u_7))$ and $|\operatorname{Spl}(f/k)| = 1536$. The time needed for its computation is **0.95 sec** and the final precision of $\operatorname{Spl}(f)$ is $\operatorname{prec}(u_6) = 548$.*


$N := k_5 = \mathbb{F}_{2^2}((u_7)) \cong k_4[X]/\langle g_6\rangle \cong k_4[X]/\langle \tilde{g}_6\rangle$
2
$k_4 := \mathbb{F}_{2^2}((u_6)) \cong k_3[X]/\langle g_4\rangle \cong k_3[X]/\langle \tilde{g}_4\rangle$
2
$k_3 := \mathbb{F}_{2^2}((u_5)) \cong k_2[X]/\langle g_2\rangle \cong k_2[X]/\langle \tilde{g}_2\rangle$
2
$k_2 := \mathbb{F}_{2^2}((u_4)) \cong k_1[X]/\langle g_1\rangle \cong k_1[X]/\langle \tilde{g}_1\rangle$
2
$KT =: k_1 := \mathbb{F}_{2^2}((u_3)) \cong T[X]/\langle f\rangle \cong T[X]/\langle g\rangle$
16
$K := k[X]/\langle f\rangle$
$T := \mathbb{F}_{2^2}((u_2))$ $f \in T[X]$
6
$k := \mathbb{F}_2((t))$ $f \in k[X]$

Given $f \in k[X]$ we compute the field T, and then we consider f over T as

$$\operatorname{Spl}(f/k) = \operatorname{Spl}(f/T).$$

Step 1: Time:* 0.44 *sec
The polynomial $f \in T[X]$ is not Eisenstein, and so we take its corresponding Eisenstein polynomial g to generate the extension k_1/T as $\operatorname{Spl}(f/T) = \operatorname{Spl}(g/T)$. Thus, the field $k_1 := \mathbb{F}_{2^2}((u_3))$ with $\operatorname{prec}(u_3) = 248$. At this point we use the formula $\pi_0 \in T[X]$, which gives the relation between the roots of g and f in order to compute the formula $B_0(X) \in T[X]$. So we store them $\langle g, B_0\rangle$ and proceed to the next step. Now we factorize g and get

$$g = (X - \tilde{\alpha}_1)(X - \tilde{\alpha}_2)\prod_{j=1}^{7} g_j \in k_1[X],$$

with $\deg(g_j) = 2$ *for* $1 \leq j \leq 7$. *We compute the roots of* f *which are*

$$\alpha_i = B_0(\tilde{\alpha}_i) \ \textit{for} \ i = 1, 2$$

and store them. Also, for every factor g_j *we determine its corresponding Eisenstein polynomial* $\tilde{g}_j$ *and the formula* $B_{1,j}$ *by the relation* $\pi_{1,j}$ *of their roots, and then we store the factors*

$$\langle \tilde{g}_j, B_0(B_{1,j}(X)) \rangle \ \textit{for every} \ 1 \leq j \leq 7.$$

Step 2: Time:* 0.12 *sec

We choose $\tilde{g}_1$ *to construct the extension* k_2/k_1, *as*

$$\mathrm{Spl}(g_1/k_1) = \mathrm{Spl}(\tilde{g}_1/k_1).$$

Thus, the field $k_2 := \mathbb{F}_{2^2}((u_4))$ *with* $\mathrm{prec}(u_4) = 276$. *By factorizing it over* k_2 *we get that*

$$\tilde{g}_1 = (X - \tilde{\alpha}_3)(X - \tilde{\alpha}_4) \in k_2[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,1}(\tilde{\alpha}_i)) \ \textit{for} \ i = 3, 4$$

and store them. In addition, we factorize and upgrade the old factors which are of the following form

$$\langle \tilde{\tilde{g}}_j, B_0(B_{1,j}(B_{2,j}(X))) \rangle \ \textit{for every} \ 2 \leq j \leq 7.$$

Step 3: Time:* 0.12 *sec

We choose $\tilde{\tilde{g}}_2$ *to construct the extension* k_3/k_2, *as*

$$\mathrm{Spl}(g_2/k_2) = \mathrm{Spl}(\tilde{g}_2/k_2) \cong \mathrm{Spl}(\tilde{\tilde{g}}_2/k_2).$$

Thus, the field $k_3 := \mathbb{F}_{2^2}((u_5))$ *with* $\mathrm{prec}(u_5) = 306$. *By factorizing it over* k_3 *we get that*

$$\tilde{\tilde{g}}_2 = (X - \tilde{\alpha}_5)(X - \tilde{\alpha}_6) \in k_3[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,2}(B_{2,2}(\tilde{\alpha}_i))) \ \textit{for} \ i = 5, 6$$

and store them. In addition, we factorize and upgrade the old factors. In this case, it holds that $\tilde{\tilde{g}}_3 \in k_3[X]$ *splits, and thus*

$$\tilde{\tilde{g}}_3 = (X - \tilde{\alpha}_7)(X - \tilde{\alpha}_8) \in k_3[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,3}(B_{2,3}(\tilde{\alpha}_i))) \ \textit{for} \ i = 7, 8$$

and store them as well. The rest factors are of the following form

$$\langle g_j^{(3)}, B_0(B_{1,j}(B_{2,j}(B_{3,j}(X)))) \rangle \ \textit{for every} \ 4 \leq j \leq 7.$$

Step 4: Time:* 0.11 *sec

We choose $g_4^{(3)}$ *to construct the extension* k_4/k_3, *as*

$$\mathrm{Spl}(g_4/k_3) = \mathrm{Spl}(\tilde{g}_4/k_3) \cong \mathrm{Spl}(\tilde{\tilde{g}}_4/k_3) \cong \mathrm{Spl}(g_4^{(3)}/k_3).$$

Thus, the field $k_4 := \mathbb{F}_{2^2}((u_6))$ *with* $\mathrm{prec}(u_6) = 356$. *By factorizing it over* k_4 *we get that*

$$g_4^{(3)} = (X - \tilde{\alpha}_9)(X - \tilde{\alpha}_{10}) \in k_4[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,4}(B_{2,4}(B_{3,4}(\tilde{\alpha}_i))) \ \textit{for} \ i = 9, 10$$

and store them. In addition, we factorize and upgrade the old factors. In this case, it holds that $g_5^{(3)} \in k_4[X]$ *splits, and thus*

$$g_5^{(3)} = (X - \tilde{\alpha}_{11})(X - \tilde{\alpha}_{12}) \in k_4[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,5}(B_{2,5}(B_{3,5}(\tilde{\alpha}_i)))) \ \textit{for} \ i = 11, 12$$

and store them as well. The rest factors are of the following form

$$\langle g_j^{(4)}, B_0(B_{1,j}(B_{2,j}(B_{3,j}(B_{4,j}(X))))) \rangle \ \textit{for every} \ 6 \leq j \leq 7.$$

Step 5: Time:* 0.06 *sec

We choose $g_6^{(4)}$ *to construct the extension* k_5/k_4, *as*

$$\mathrm{Spl}(g_6/k_4) = \mathrm{Spl}(\tilde{g}_6/k_4) \cong \mathrm{Spl}(\tilde{\tilde{g}}_6/k_3) \cong \mathrm{Spl}(g_6^{(3)}/k_4) \cong \mathrm{Spl}(g_6^{(4)}/k_4).$$

Thus, the field $k_5 := \mathbb{F}_{2^2}((u_7))$ *with* $\mathrm{prec}(u_7) = 548$. *By factorizing it over* k_5 *we get that*

$$g_6^{(4)} = (X - \tilde{\alpha}_{13})(X - \tilde{\alpha}_{14}) \in k_5[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,6}(B_{2,6}(B_{3,6}(B_{4,6})(\tilde{\alpha}_i)))) \text{ for } i = 13, 14$$

and store them. In addition, we factorize and upgrade the old factors. In this case, it holds that $g_7^{(4)} \in k_5[X]$ *splits, and thus*

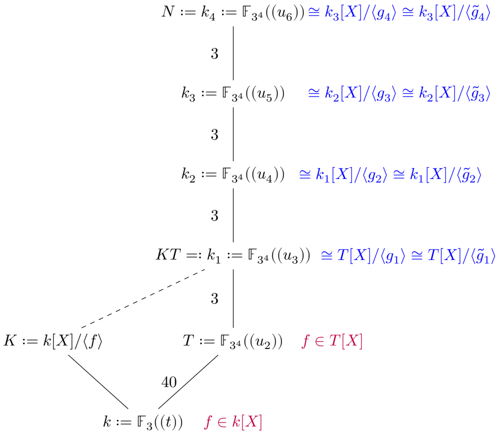
$$g_7^{(4)} = (X - \tilde{\alpha}_{15})(X - \tilde{\alpha}_{16}) \in k_5[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,7}(B_{2,7}(B_{3,7}(B_{4,7}(\tilde{\alpha}_i)))) \text{ for } i = 15, 16$$

and store them as well.
The polynomial f *factors completely, which completes its splitting field computation.*

Example 5.4.3
Let $f(X) = X^{15} + tX^3 + tX + t \in k[X]$, *where* $k := \mathbb{F}_3((t))$. *Then,* $\mathrm{Spl}(f/k) = \mathbb{F}_{3^4}((u_6))$ *and* $|\mathrm{Spl}(f/k)| = 3240$. *The time needed for its computation is* ***4.5 sec*** *and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(u_6) = 830$.


$N := k_4 := \mathbb{F}_{3^4}((u_6)) \cong k_3[X]/\langle g_4 \rangle \cong k_3[X]/\langle \tilde{g}_4 \rangle$
3
$k_3 := \mathbb{F}_{3^4}((u_5)) \cong k_2[X]/\langle g_3 \rangle \cong k_2[X]/\langle \tilde{g}_3 \rangle$
3
$k_2 := \mathbb{F}_{3^4}((u_4)) \cong k_1[X]/\langle g_2 \rangle \cong k_1[X]/\langle \tilde{g}_2 \rangle$
3
$KT =: k_1 := \mathbb{F}_{3^4}((u_3)) \cong T[X]/\langle g_1 \rangle \cong T[X]/\langle \tilde{g}_1 \rangle$
3
$K := k[X]/\langle f \rangle$
$T := \mathbb{F}_{3^4}((u_2))$ $f \in T[X]$
40
$k := \mathbb{F}_3((t))$ $f \in k[X]$

Given $f \in k[X]$ *we compute the field* T, *where* $t = u_2^{10}$, *and then we consider* f *over* T *as*

$$\mathrm{Spl}(f/k) = \mathrm{Spl}(f/T).$$

Step 1: Time:* 0.25 *sec
The polynomial $f(X) = X^{15} + u_2^{10}X^3 + u_2^{10}X + u_2^{10} \in T[X]$ *is reducible and by factorizing it we get*

$$f = g_1 g_2 g_3 g_4 g_5 \in T[X], \text{ with } \deg(g_j) = 3 \text{ for } 1 \leq j \leq 5.$$

For every factor g_j *we determine its corresponding Eisenstein polynomial* $\tilde{g}_j$ *and the formula* $B_{0,j}$ *by the relation* $\pi_{1,j}$ *of their roots, and then we store the factors*

$$\langle \tilde{g}_j, B_{0,j}(X) \rangle \text{ for every } 1 \leq j \leq 5.$$

We choose $\tilde{g}_1$ *to construct the extension* k_1/T, *as* $\mathrm{Spl}(g_1/T) = \mathrm{Spl}(\tilde{g}_1/T)$. *Thus, the field* $k_1 := \mathbb{F}_{3^4}((u_3))$ *with* $\mathrm{prec}(u_3) = 268$. *By factorizing it over* k_1 *we get that*

$$\tilde{g}_1 = \prod_{i=1}^{3}(X - \tilde{\alpha}_i) \in k_1[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_{0,1}(\tilde{\alpha}_i)), \text{ for } i = 1, 2, 3$$

and store them. In addition to it, we factorize and upgrade the old factors which are of the following form

$$\langle \tilde{\tilde{g}}_j, B_{0,j}(B_{1,j}(X)) \rangle \text{ for every } 2 \leq j \leq 5.$$

Step 2: Time:* 0.45 *sec
We choose $\tilde{\tilde{g}}_2$ *to construct the extension* k_2/k_1, *as*

$$\mathrm{Spl}(g_2/k_1) = \mathrm{Spl}(\tilde{g}_2/k_1) = \mathrm{Spl}(\tilde{\tilde{g}}_2/k_1).$$

Thus, the field $k_2 := \mathbb{F}_{3^4}((u_4))$ *with* $\operatorname{prec}(u_4) = 310$. *By factorizing it over* k_2 *we get that*

$$\tilde{\tilde{g}}_2 = \prod_{i=4}^{6}(X - \tilde{\alpha}_i) \in k_2[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_{0,2}(B_{1,2}(\tilde{\alpha}_i)) \text{ for } i = 4, 5, 6$$

and store them. In addition to it, we factorize and upgrade the old factors which are of the following form

$$\langle g_j^{(3)}, B_{0,j}(B_{1,j}(B_{2,j}(X)))\rangle \text{ for every } 3 \le j \le 5.$$

Step 3: Time:* 0.71 *sec

We choose $g_3^{(3)}$ *to construct the extension* k_3/k_2, *as*

$$\operatorname{Spl}(g_3/k_2) = \operatorname{Spl}(\tilde{g}_3/k_2) = \operatorname{Spl}(\tilde{\tilde{g}}_3/k_2) = \operatorname{Spl}(g_3^{(3)}/k_3).$$

Thus, the field $k_3 := \mathbb{F}_{3^4}((u_5))$ *with* $\operatorname{prec}(u_5) = 412$. *By factorizing it over* k_3 *we get that*

$$g_3^{(3)} = \prod_{i=7}^{9}(X - \tilde{\alpha}_i) \in k_3[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_{0,3}(B_{1,3}(B_{2,3}(\tilde{\alpha}_i))) \text{ for } i = 7, 8, 9$$

and store them. In addition, we factorize and upgrade the old factors which are of the following form

$$\langle g_j^{(4)}, B_0(B_{1,j}(B_{2,j}(B_{3,j}(X))))\rangle \text{ for every } 4 \le j \le 5.$$

Step 4: Time:* 2.47 *sec

We choose $g_4^{(4)}$ *to construct the extension* k_4/k_3, *as*

$$\operatorname{Spl}(g_4/k_3) = \operatorname{Spl}(\tilde{g}_4/k_3) \cong \operatorname{Spl}(\tilde{\tilde{g}}_4/k_3) \cong \operatorname{Spl}(g_4^{(3)}/k_3) \cong \operatorname{Spl}(g_4^{(4)}/k_3).$$

Thus, the field $k_4 := \mathbb{F}_{3^4}((u_6))$ *with* $\operatorname{prec}(u_6) = 830$. *By factorizing it over* k_4 *we get that*

$$g_4^{(4)} = (X - \tilde{\alpha}_{10})(X - \tilde{\alpha}_{11})(X - \tilde{\alpha}_{12}) \in k_4[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_{0,4}(B_{1,4}(B_{2,4}(B_{3,4}(\tilde{\alpha}_i))) \text{ for } i = 4, 5, 6$$

and store them. In addition, we factorize and upgrade the old factor $g_5^{(4)} \in k_4[X]$ *which splits, and thus*

$$g_5^{(4)} = (X - \tilde{\alpha}_{13})(X - \tilde{\alpha}_{14})(X - \tilde{\alpha}_{15}) \in k_4[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_{0,5}(B_{1,5}(B_{2,5}(B_{3,5}(\tilde{\alpha}_i))) \text{ for } i = 13, 14, 15$$

and store them as well.

The polynomial f *factors completely, which completes its splitting field computation.*

5.5 Examples: Comparison of Version 2 to Version 3

Having explained in theory this approach, it is of interest to examine its effect in practice. To do this, we will thoroughly look into some examples in which we will compare the aforementioned approach with the so-called "Combined Approach". Some relevant examples are indicated.

Example 5.5.1

Let $f(X) = X^{16} + t^5X^{10} + t^7X^9 + t^2X^8 + tX^4 + t \in k[X]$, *where* $k := \mathbb{F}_2((t))$. *The polynomial* f *is an Eisenstein polynomial and its ramification polygon consists of two segments with absolute value of its slopes* $[101/3, 1/3]$. *Having computed its splitting field we know that*

$$\operatorname{Spl}(f/k) = \mathbb{F}_{2^2}((u_6)) \text{ and } |\operatorname{Spl}(f/k)| = 6144 = 2^{11} \cdot 3.$$

Also, we get access to its roots, and so we can determine the u_4*-valuation of its roots, which is given below:*

$$v_{u_4}(\alpha_i - \alpha) := \{v_{u_4}(\alpha_i - \alpha) \ : \ 1 \le i \le 16\} = \{256^{12}, 6656^3, \infty\}.$$

$$\begin{tikzcd}[column sep=small]
 & \tilde{k}_3[X]/\langle f_3\rangle \cong \mathbb{F}_{2^2}((\tilde{u}_6)) =: \tilde{k}_4 \cong N \cong k_4 := \mathbb{F}_{2^2}((u_6)) \cong k_3[X]/\langle g_3\rangle \cong k_3[X]/\langle g_3^{(3)}\rangle \arrow[dl, dash, "4"'] \arrow[dr, dash, "4"] & \\
\tilde{k}_2[X]/\langle f_2\rangle \cong \tilde{k}_3 := \mathbb{F}_{2^2}((\tilde{u}_5)) \arrow[d, dash, "4"'] & & k_3 := \mathbb{F}_{2^2}((u_5)) \cong k_2[X]/\langle g_2\rangle \cong k_2[X]/\langle \tilde{\tilde{g}}_2\rangle \arrow[d, dash, "4"] \\
\tilde{k}_1[X]/\langle f_1\rangle \cong \tilde{k}_2 := \mathbb{F}_2((\tilde{u}_4)) \arrow[d, dash, "4"'] & & k_2 := \mathbb{F}_{2^2}((u_4)) \cong k_1[X]/\langle g_1\rangle \cong k_1[X]/\langle \tilde{g}_1\rangle \arrow[d, dash, "4"] \\
T[X]/\langle f\rangle \cong \tilde{k}_1 := \mathbb{F}_2((\tilde{u}_3)) \arrow[dr, dash, "16"'] & & k_1 := \mathbb{F}_{2^2}((u_3)) \cong T[X]/\langle f\rangle \cong T[X]/\langle g\rangle \arrow[dl, dash, "16"] \\
 & T := \mathbb{F}_{2^2}((u_2)) \quad f \in T[X] \arrow[d, dash, "6"] & \\
 & k := \mathbb{F}_2((t)) \quad f \in k[X] &
\end{tikzcd}$$

Computing $\mathrm{Spl}(f)$ *by using the so-called "Combined Approach", the time needed for its computation is* ***8227.4 sec*** $\sim$ ***2.3h*** *and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(u_6) = 43988$. *On the other hand, we employ the splitting field algorithm based on the so-called "Improved Approach" for its determination. In that case, though, the time needed for its computation is* ***65.79 sec*** *and the final precision of* $\mathrm{Spl}(f)$ *is* $\mathrm{prec}(u_6) = 6932$.

We begin with computing $\mathrm{Spl}(f)$ *by using the so-called "Combined Approach". Let us give its description step by step. Given* $f \in k[X]$ *we compute the field* T *which is* $T := \mathbb{F}_{2^2}((u_2))$ *with respect to the the relation* $\Phi_2(\Phi_1(t)) = u_2^3$ *of degree* $[T : k] = 6 = (2) \cdot (3)$ *and* $\mathrm{prec}(u_2) = 948$, *and then we consider* f *over* T *as*

$$\mathrm{Spl}(f/k) = \mathrm{Spl}(f/T).$$

By factorizing it over T, *we get that* f *is irreducible.*

Step 1: Time:* 21.21 *sec

We take the polynomial $f \in T[X]$ *to generate the extension* $\tilde{k}_1/T$. *Thus, the field* $\tilde{k}_1 := \mathbb{F}_{2^2}((\tilde{u}_3))$ *with* $\mathrm{prec}(\tilde{u}_3) = 2704$. *Now we factorize* f *and get*

$$f = \prod_{i=1}^{4}(X - \alpha_i) f_1 f_2 f_3 \in \tilde{k}_1[X], \quad \textit{with} \quad \deg(f_j) = 4 \ \forall j = 1, 2, 3.$$

We upgrade the stored factors of f *so as to be over* $\tilde{k}_1[X]$. *So, we store the irreducible factors as well as the roots of* f *and proceed to the next step.*

Step 2: Time:* 492.04 *sec $\sim$ **8.2*min***

We choose f_1 *to construct the extension* $\tilde{k}_2/\tilde{k}_1$. *Thus, the field* $\tilde{k}_2 := \mathbb{F}_{2^2}((\tilde{u}_4))$ *with* $\mathrm{prec}(\tilde{u}_4) = 10944$. *We consider* f_1 *over* $\tilde{k}_2$ *and by factorizing it over* $\tilde{k}_2$ *we get that*

$$f_1 = \prod_{i=5}^{8}(X - \alpha_i) \in \tilde{k}_2[X].$$

We upgrade the stored factors of f *so as to be over* $\tilde{k}_2[X]$. *It holds that* $f_2, f_3 \in \tilde{k}_2[X]$ *are still irreducible. So, we store them as well as the roots of* f *and proceed to the next step.*

Step 3: Time:* 3340.16 *sec $\sim$ **55.6*min***

We choose f_2 *to construct the extension* $\tilde{k}_3/\tilde{k}_2$. *Thus, the field* $\tilde{k}_3 := \mathbb{F}_{2^2}((\tilde{u}_5))$ *with* $\mathrm{prec}(\tilde{u}_5) = 43776$. *We consider* f_2 *over* $\tilde{k}_3$ *and by factorizing it over* $\tilde{k}_3$ *we get that*

$$f_2 = \prod_{i=9}^{12}(X - \alpha_i) \in \tilde{k}_3[X].$$

We upgrade the stored factors of f *so as to be over* $\tilde{k}_3[X]$. *It holds that* $f_3 \in \tilde{k}_3[X]$ *is still irreducible. So, we store it as well as the roots of* f *and proceed to the next step.*

Step 4: Time:* 3888.85 *sec $\sim$ **64*min***

We choose f_3 *to construct the extension* $\tilde{k}_4/\tilde{k}_3$. *Thus, the field* $\tilde{k}_4 := \mathbb{F}_{2^2}((\tilde{u}_6))$ *with* $\mathrm{prec}(\tilde{u}_6) = 43988$. *We consider* f_3 *over* $\tilde{k}_4$ *and by factorizing it over* $\tilde{k}_4$ *we get that*

$$f_3 = \prod_{i=13}^{16}(X - \alpha_i) \in \tilde{k}_4[X].$$

The polynomial f factors completely, which completes its splitting field computation.

Let us now describe the splitting field algorithm based on the so-called "Improved Approach" in detail. Given $f \in k[X]$ we compute the field T which is $T := \mathbb{F}_{2^2}((u_2))$ with respect to the relation $\Phi_2(\Phi_1(t)) = u_2^3$ of degree $[T : k] = 6 = (2) \cdot (3)$ and prec$(u_2) = 948$, *and then we consider f over T as*

$$\mathrm{Spl}(f/k) = \mathrm{Spl}(f/T).$$

By factorizing it over T, we get that f is irreducible.

Step 1: Time:* 12.82 *sec

The polynomial $f \in T[X]$ is not Eisenstein, and so we take its corresponding Eisenstein polynomial g to generate the extension k_1/T as

$$\mathrm{Spl}(f/T) = \mathrm{Spl}(g/T).$$

Thus, the field $k_1 := \mathbb{F}_{2^2}((u_3))$ with prec$(u_3) = 2038$. *At this point we use the formula $\pi_0 \in T[X]$, which gives the relation between the roots of g and f in order to compute the formula $B_0(X) \in T[X]$. So we store them $\langle g, B_0 \rangle$ and proceed to the next step. Now we factorize g and get*

$$g = \prod_{i=1}^{4} (X - \tilde{\alpha}_i) g_1 g_2 g_3 \in k_1[X],$$

with $\deg(g_j) = 4$ *for $1 \leq j \leq 3$. We compute the roots of f which are*

$$\alpha_i = B_0(\tilde{\alpha}_i) \ \ for\ \ 1 \leq i \leq 4$$

and store them. Also, for every factor g_j we determine its corresponding Eisenstein polynomial $\tilde{g}_j$ and the formula $B_{1,j}$ by the relation $\pi_{1,j}$ of their roots, and then we store the factors

$$\langle \tilde{g}_j, B_0(B_{1,j}(X)) \rangle \ \ for\ every\ \ 1 \leq j \leq 3.$$

So, we store the irreducible factors as well as the roots of f and proceed to the next step.

Step 2: Time:* 28.13 *sec

We choose $\tilde{g}_1$ to construct the extension k_2/k_1, as

$$\mathrm{Spl}(g_1/k_1) = \mathrm{Spl}(\tilde{g}_1/k_1).$$

Thus, the field $k_2 := \mathbb{F}_{2^2}((u_4))$ with prec$(u_4) = 2137$. *By factorizing it over k_2 we get that*

$$\tilde{g}_1 = \prod_{i=5}^{8} (X - \tilde{\alpha}_i) \in k_2[X].$$

We determine the roots of f which are

$$\alpha_i = B_0(B_{1,1}(\tilde{\alpha}_i)) \ \ for\ \ 5 \leq i \leq 8$$

and store them. In addition, we upgrade the old factors so as to be over $k_2[X]$ and we factorize them. It holds that $g_2, g_3 \in k_2[X]$ are still irreducible. So, we store them in the following form

$$\langle \tilde{\tilde{g}}_j, B_0(B_{1,j}(B_{2,j}(X))) \rangle \ \ for\ every\ \ 2 \leq j \leq 3,$$

and proceed to the next step.

Step 3: Time:* 16.15 *sec

We choose $\tilde{\tilde{g}}_2$ to construct the extension k_3/k_2, as

$$\mathrm{Spl}(g_2/k_2) = \mathrm{Spl}(\tilde{g}_2/k_2) \cong \mathrm{Spl}(\tilde{\tilde{g}}_2/k_2).$$

Thus, the field $k_3 := \mathbb{F}_{2^2}((u_5))$ with prec$(u_5) = 2266$. *By factorizing it over k_3 we get that*

$$\tilde{\tilde{g}}_2 = \prod_{i=9}^{12} (X - \tilde{\alpha}_i) \in k_3[X].$$

We determine the roots of f which are

$$\alpha_i = B_0(B_{1,2}(B_{2,2}(\tilde{\alpha}_i))) \ \ for\ \ 9 \leq i \leq 12$$

and store them. In addition, we factorize and upgrade the old factors so as to be over $k_3[X]$. It holds that $g_3 \in k_3[X]$ is still irreducible. So, we store them in the following form

$$\langle g_3^{(3)}, B_0(B_{1,3}(B_{2,3}(B_{3,3}(X)))) \rangle,$$

and proceed to the next step.

Step 4: Time:* 6.71 *sec

We choose $g_3^{(3)}$ to construct the extension k_4/k_3, as

$$\mathrm{Spl}(g_3/k_3) = \mathrm{Spl}(\tilde{g}_3/k_3) \cong \mathrm{Spl}(\tilde{\tilde{g}}_3/k_3) \cong \mathrm{Spl}(g_3^{(3)}/k_3).$$

Thus, the field $k_4 := \mathbb{F}_{2^2}((u_6))$ *with* $\operatorname{prec}(u_6) = 6932$. *By factorizing it over* k_4 *we get that*

$$g_3^{(3)} = \prod_{i=13}^{16} (X - \tilde{\alpha}_i)) \in k_4[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,3}(B_{2,3}(B_{3,3}(\tilde{\alpha}_i))) \text{ for } 13 \leq i \leq 16$$

and store them.
The polynomial f *factors completely, which completes its splitting field computation.*

Example 5.5.2
Let $f(X) = X^{16} + t^5X^{14} + t^7X^{13} + t^2X^{12} + tX^8 + t \in k[X]$, *where* $k := \mathbb{F}_2((t))$. *The polynomial* f *is an Eisenstein polynomial and its ramification polygon consists of four segments with absolute value of its slopes* $[31, 25, 5, 1]$. *Having computed its splitting field we know that*

$$\operatorname{Spl}(f/k) = \mathbb{F}_{2^2}((u_6)) \text{ and } |\operatorname{Spl}(f/k)| = 2048 = 2^{11}.$$

Also, we get access to its roots, and so we can determine the u_6*-valuation of its roots, which is given below:*

$$v_{u_6}(\alpha_i - \alpha) := \{v_{u_6}(\alpha_i - \alpha) \ : \ 1 \leq i \leq 16\} = \{128^8, 384^4, 1664^2, 2048, \infty\}.$$

$$\begin{tikzcd}
 & \tilde{k}_5[X]/\langle f_{3,2}^1\rangle \cong \mathbb{F}_{2^2}((\tilde{u}_6)) =: \tilde{k}_6 \cong N \cong k_6 := \mathbb{F}_{2^2}((u_6)) \cong k_5[X]/\langle g_{2,1}^{(4)}\rangle \cong k_5[X]/\langle g_{2,1}^{(5)}\rangle \arrow[dl, dash, "2"'] \arrow[dr, dash, "2"] & \\
\tilde{k}_4[X]/\langle f_{3,1}\rangle \cong \tilde{k}_5 := \mathbb{F}_{2^2}((\tilde{u}_5)) \arrow[d, dash, "4"'] & & k_5 := \mathbb{F}_{2^2}((u_5)) \cong k_4[X]/\langle g_{3,2}^{(3)}\rangle \cong k_4[X]/\langle g_{3,2}^{(4)}\rangle \arrow[d, dash, "4"] \\
\tilde{k}_3[X]/\langle f_2\rangle \cong \tilde{k}_4 := \mathbb{F}_{2^2}((\tilde{u}_4)) \arrow[d, dash, "4"'] & & k_4 := \mathbb{F}_{2^2}((u_4)) \cong k_3[X]/\langle \tilde{\tilde{g}}_{3,1,1}\rangle \cong k_3[X]/\langle g_{3,1,1}^{(3)}\rangle \arrow[d, dash, "2"] \\
\tilde{k}_2[X]/\langle f_1\rangle \cong \tilde{k}_3 := \mathbb{F}_{2^2}((\tilde{u}_3)) \arrow[d, dash, "2"'] & & k_3 := \mathbb{F}_{2^2}((u_3)) \cong k_2[X]/\langle \tilde{g}_{3,1}\rangle \cong k_2[X]/\langle \tilde{\tilde{g}}_{3,1}\rangle \arrow[d, dash, "4"] \\
\tilde{k}_2 := \mathbb{F}_2((\tilde{u}_2)) \arrow[d, dash, "2"'] & & k_2 := \mathbb{F}_{2^2}((u_2)) \arrow[d, dash, "2"] \\
T[X]/\langle f\rangle \cong \tilde{k}_1 := \mathbb{F}_2((\tilde{u}_1)) \arrow[dr, dash, "16"'] & & k_1 := \mathbb{F}_2((u_1)) \cong T[X]/\langle f\rangle \cong T[X]/\langle g\rangle \arrow[dl, dash, "16"] \\
 & T := \mathbb{F}_2((t)) \quad f \in T[X] \arrow[d, dash, "1"] & \\
 & k := \mathbb{F}_2((t)) \quad f \in k[X] &
\end{tikzcd}$$

Computing $\operatorname{Spl}(f)$ *by using the so-called "Combined Approach", the time needed for its computation is* ***725.15 sec*** $\sim$ ***12min*** *and the final precision of* $\operatorname{Spl}(f)$ *is* $\operatorname{prec}(u_6) = 23636$. *On the other hand, we employ the splitting field algorithm based on the so-called "Improved Approach" for its determination. In that case, though, the time needed for its computation is* ***60.54 sec*** *and the final precision of* $\operatorname{Spl}(f)$ *is* $\operatorname{prec}(u_6) = 2375$.

We begin with computing $\operatorname{Spl}(f)$ *by using the so-called "Combined Approach". Let us give its description step by step.*
Given $f \in k[X]$ *we compute the field* T *which is trivial, i.e* $T = k = \mathbb{F}_2((t))$ *and of degree* $[T : k] = 1$ *and then we consider* f *over* T *as* $\operatorname{Spl}(f/k) = \operatorname{Spl}(f/T)$. *By factorizing it over* T, *we get that* f *is irreducible.*

Step 1: Time: 21.54 sec
We take the polynomial $f \in T[X]$ *to generate the extension* $\tilde{k}_1/T$. *Thus, the field* $\tilde{k}_1 := \mathbb{F}_2((\tilde{u}_1))$ *with* $\operatorname{prec}(\tilde{u}_1) = 1965$. *Now we factorize* f *and get*

$$f = \prod_{i=1}^{2}(X - \alpha_i) f_1 f_2 f_3 \in \tilde{k}_1[X],$$

with $\deg(f_1) = 1$, $\deg(f_2) = 4$ *and* $\deg(f_3) = 8$. *We upgrade the stored factors of* f *so as to be over* $\tilde{k}_1[X]$. *So, we store the irreducible factors as well as the roots of* f *and proceed to the next step.*

Step 2: Time: 3.78 sec
At this step, by checking the stored irreducible factors, we get a non-trivial inertia degree, so we

have a constant field extension. Thus, the field $\tilde{k}_2 := \mathbb{F}_{2^2}((u_2))$ *with* $\mathrm{prec}(\tilde{u}_2) = 1965$. *We upgrade the stored factors of* f *so as to be over* $\tilde{k}_2[X]$. *Then, we factorize* $f_1, f_2 \in \tilde{k}_2[X]$ *and get that they are still irreducible. It is hold that* f_3 *is not irreducible, and we get that*

$$f_3 = f_{3,1} f_{3,2} \in k_2[X], \quad \textit{where} \ \deg(f_{3,i}) = 4 \textit{ for } i = 1, 2.$$

So, we store them as well as the roots of f *and proceed to the next step.*

Step 3: Time:* 23.69 *sec

We choose f_1 *to construct the extension* $\tilde{k}_3/\tilde{k}_2$. *Thus, the field* $\tilde{k}_3 := \mathbb{F}_{2^2}((\tilde{u}_3))$ *with* $\mathrm{prec}(\tilde{u}_3) = 3782$. *We consider* f_1 *over* $\tilde{k}_3$ *and by factorizing it over* $\tilde{k}_3$ *we get that*

$$f_1 = (X - \alpha_3)(X - \alpha_4) \in \tilde{k}_3[X].$$

We upgrade the stored factors of f *so as to be over* $\tilde{k}_3[X]$. *Then, we factorize them and get that they are still irreducible. So, we store them as well as the roots of* f *and proceed to the next step.*

Step 4: Time:* 85.86 *sec

We choose f_2 *to construct the extension* $\tilde{k}_4/\tilde{k}_3$. *Thus, the field* $\tilde{k}_4 := \mathbb{F}_{2^2}((\tilde{u}_4))$ *with* $\mathrm{prec}(\tilde{u}_4) = 5376$. *We consider* f_2 *over* $\tilde{k}_4$ *and by factorizing it over* k_4 *we get that*

$$f_2 = \prod_{i=5}^{8}(X - \alpha_i) \in \tilde{k}_4[X].$$

We upgrade the stored factors of f *so as to be over* $k_4[X]$. *Then, we factorize them and get that they are still irreducible. So, we store them as well as the roots of* f *and proceed to the next step.*

Step 5: Time:* 104.95 *sec* $\sim$ 1.7*min

We choose $f_{3,1}$ *to construct the extension* $\tilde{k}_5/\tilde{k}_4$. *Thus, the field* $\tilde{k}_5 := \mathbb{F}_{2^2}((\tilde{u}_5))$ *with* $\mathrm{prec}(\tilde{u}_5) = 21504$. *We consider* $f_{3,1}$ *over* $\tilde{k}_5$ *and by factorizing it over* $\tilde{k}_5$ *we get that*

$$f_{3,1} = \prod_{i=9}^{12}(X - \alpha_i) \in \tilde{k}_5[X].$$

We upgrade the stored factors of f *to be over* $\tilde{k}_5[X]$. *Then, we factorize* $f_{3,2} \in \tilde{k}_5[X]$ *and get that*

$$f_{3,2} = f_{3,2}^1 f_{3,2}^2 \in \tilde{k}_5[X], \quad \textit{where} \ \deg(f_{3,2}^1) = 2 \textit{ for } i = 1, 2.$$

So, we store them as well as the roots of f *and proceed to the next step.*

Step 6: Time:* 371.80 *sec* $\sim$ 6.1*min

We choose $f_{3,2}^1$ *to construct the extension* $\tilde{k}_6/\tilde{k}_5$. *Thus, the field* $k_6 := \mathbb{F}_{2^2}((\tilde{u}_6))$ *with* $\mathrm{prec}(\tilde{u}_6) = 23636$. *We consider* $f_{3,2}$ *over* $\tilde{k}_6$ *and by factorizing it over* $\tilde{k}_6$ *we get that*

$$f_{3,2}^1 = \prod_{i=13}^{14}(X - \alpha_i) \in \tilde{k}_6[X].$$

We upgrade the stored factors of f *to be over* $\tilde{k}_6[X]$. *Then, we factorize* $f_{3,2}^2 \in \tilde{k}_6[X]$ *and get that*

$$f_{3,2}^2 = \prod_{i=15}^{16}(X - \alpha_i) \in \tilde{k}_6[X].$$

The polynomial f *factors completely, which completes its splitting field computation.*

Let us describe the splitting field algorithm based on the so-called "Improved Approach" in detail. Given $f \in k[X]$ *we compute the field* T *which is trivial, i.e* $T = k = \mathbb{F}_2((t))$ *and of degree* $[T : k] = 1$ *and then we consider* f *over* T *as* $\mathrm{Spl}(f/k) = \mathrm{Spl}(f/T)$. *By factorizing it over* T, *we get that* f *is irreducible.*

Step 1: Time:* 25.54 *sec

The polynomial $f \in T[X]$ *is Eisenstein, and so* $g = f$ *generates the extension* k_1/T *as*

$$\mathrm{Spl}(f/T) = \mathrm{Spl}(g/T).$$

Thus, the field $k_1 := \mathbb{F}_2((u_1))$ *with* $\mathrm{prec}(u_1) = 1965$. *At this point we use the formula* $\pi_0 \in T[X]$, *which gives the relation between the roots of* g *and* f *in order to compute the formula* $B_0(X) \in T[X]$. *In this case* $B_0(X) = X$ *as* $f = g$. *So we store them* $\langle g, B_0 \rangle$ *and proceed to the next step. Now we factorize* g *and get*

$$g = \prod_{i=1}^{2}(X - \tilde{\alpha}_i) g_1 g_2 g_3 \in k_1[X],$$

with $\deg(g_1) = 2$, $\deg(g_2) = 4$ *and* $\deg(g_3) = 8$. *We compute the roots of f which are*

$$\alpha_i = B_0(\tilde{\alpha}_i) \text{ for } 1 \leq i \leq 2$$

and store them. Also, for every factor g_j we determine its corresponding Eisenstein polynomial $\tilde{g}_j$ and the formula $B_{1,j}$ by the relation $\pi_{1,j}$ of their roots, and then we store the factors

$$\langle \tilde{g}_j, B_0(B_{1,j}(X)) \rangle \text{ for every } 1 \leq j \leq 3.$$

So, we store the irreducible factors as well as the roots of f and proceed to the next step.

Step 2: Time:* 3.22 *sec

At this step, by checking the stored irreducible factors, we get a non-trivial inertia degree, so we have that k_2/k_1 constant field extension. Thus, the field $k_2 := \mathbb{F}_{2^2}((u_2))$ with $\text{prec}(\tilde{u}_2) = 1965$. *We upgrade the stored factors of f so as to be over $k_2[X]$. Then, we factorize $\tilde{g}_1, \tilde{g}_2 \in k_2[X]$ and get that they are still irreducible. It is hold that $\tilde{g}_3$ is not irreducible, and we get that*

$$\tilde{g}_3 = \tilde{g}_{3,1}\tilde{g}_{3,2} \in k_2[X], \quad \text{where } \deg(f_{3,i}) = 4 \text{ for } i = 1,2.$$

So, we store them in the following form

$$\langle \tilde{\tilde{g}}_j, B_0(B_{1,j}(B_{2,j}(X))) \rangle \text{ for every } 1 \leq j \leq 2$$

and

$$\langle \tilde{\tilde{g}}_{3,j}, B_0(B_{1,3}(B_{2,3,j}(X))) \rangle \text{ for every } 1 \leq j \leq 2$$

and proceed to the next step.

Step 3: Time:* 10.75 *sec

We choose $\tilde{\tilde{g}}_{3,1}$ to construct the extension k_3/k_2, as

$$\text{Spl}(\tilde{g}_{3,1}/k_2) \cong \text{Spl}(\tilde{\tilde{g}}_{3,1}/k_2).$$

Thus, the field $k_3 := \mathbb{F}_{2^2}((u_3))$ with $\text{prec}(u_3) = 2182$. *By factorizing it over k_3 we get that*

$$\tilde{\tilde{g}}_{3,1} = \prod_{i=3}^{4}(X - \tilde{\alpha}_i)\tilde{\tilde{g}}_{3,1,1} \in k_3[X].$$

We determine the roots of f which are

$$\alpha_i = B_0(B_{1,3}(B_{2,3,1}(\tilde{\alpha}_i))) \text{ for } 3 \leq i \leq 4$$

and store them. In addition, we factorize and upgrade the old factors so as to be over $k_3[X]$. It holds that they are still irreducible. So, we store them in the following form

$$\langle g_j^{(3)}, B_0(B_{1,j}(B_{2,j}(B_{3,j}(X)))) \rangle \text{ for every } 1 \leq j \leq 2,$$

$$\langle g_{3,2}^{(3)}, B_0(B_{1,3}(B_{2,3,2}(B_{3,2}^{(3)}(X)))) \rangle$$

and

$$\langle g_{3,1,1}^{(3)}, B_0(B_{1,3}(B_{2,3,1}(B_{3,1,1}^{(3)}(X)))) \rangle$$

and proceed to the next step.

Step 4: Time:* 9.24 *sec

We choose $g_{3,1,1}^{(3)}$ to construct the extension k_4/k_3, as

$$\text{Spl}(\tilde{\tilde{g}}_{3,1,1}/k_3) \cong \text{Spl}(g_{3,1,1}^{(3)}/k_3).$$

Thus, the field $k_4 := \mathbb{F}_{2^2}((u_6))$ with $\text{prec}(u_4) = 2202$. *By factorizing it over k_4 we get that*

$$g_{3,1,1}^{(3)} = \prod_{i=5}^{6}(X - \tilde{\alpha}_i)) \in k_4[X].$$

We determine the roots of f which are

$$\alpha_i = B_0(B_{1,3}(B_{2,3,1}(B_{3,1,1}^{(3)}(\tilde{\alpha}_i))) \text{ for } 5 \leq i \leq 6$$

and store them. In addition, we factorize and upgrade the old factors so as to be over $k_4[X]$. It is hold that $g_1^{(3)}$ is not irreducible, and we get that

$$g_1^{(3)} = \prod_{i=7}^{8}(X - \tilde{\alpha}_i)) \in k_4[X].$$

We determine the roots of f which are

$$\alpha_i = B_0(B_{1,1}(B_{2,1}(B_{3,1}(\tilde{\alpha}_i))) \text{ for } 7 \leq i \leq 8$$

and store them. The rest are still irreducible. So, we store them in the following form

$$\langle g_2^{(4)}, B_0(B_{1,2}(B_{2,2}(B_{3,2}(B_{4,2}(X))))) \rangle$$

and

$$\langle g_{3,2}^{(4)}, B_0(B_{1,3}(B_{2,3,2}(B_{3,2}^{(3)}(B_{3,2}^{(4)}(X))))) \rangle,$$

and proceed to the next step.
Step 5: Time: **6.98** ***sec***
We choose $g^{(4)}_{3,2}$ *to construct the extension* k_5/k_4*, as*

$$\operatorname{Spl}(\tilde{g}_{3,2}/k_4) \cong \operatorname{Spl}(\tilde{\tilde{g}}_{3,2}/k_4) \cong \operatorname{Spl}(g^{(3)}_{3,2}/k_4) \cong \operatorname{Spl}(g^{(4)}_{3,2}/k_4).$$

Thus, the field $k_5 := \mathbb{F}_{2^2}((u_5))$ *with* $\operatorname{prec}(u_5) = 2277$*. By factorizing it over* k_5 *we get that*

$$g^{(4)}_{3,2} = \prod_{i=9}^{12}(X - \tilde{\alpha}_i)) \in k_5[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,3}(B_{2,3,2}(B^{(3)}_{3,2}(B^{(4)}_{3,2}(\tilde{\alpha}_i)))) \text{ for } 9 \leq i \leq 12$$

and store them. In addition, we factorize and upgrade the old factors so as to be over $k_5[X]$*. It is hold that* $g^{(4)}_2$ *is not irreducible, and we get that*

$$g^{(4)}_2 = g^{(4)}_{2,1} g^{(4)}_{2,2} \in k_5[X], \text{ with } \deg(g^{(4)}_{2,i}) = 2 \text{ for } i = 1, 2.$$

So, we store it in the following form

$$\langle g^{(5)}_{2,j}, B_0(B_{1,2}(B_{2,2}(B_{3,2}(B_{4,2}(B_{5,2,j}(X))))))\rangle, \text{ for } j = 1, 2$$

and proceed to the next step.
Step 6: Time: **3.37** ***sec***
We choose $g^{(5)}_{2,1}$ *to construct the extension* k_6/k_5*, as*

$$\operatorname{Spl}(g^{(4)}_{2,1}/k_5) \cong \operatorname{Spl}(g^{(5)}_{2,1}/k_5).$$

Thus, the field $k_6 := \mathbb{F}_{2^2}((u_6))$ *with* $\operatorname{prec}(u_6) = 2375$*. By factorizing it over* k_6 *we get that*

$$g^{(5)}_{2,1} = \prod_{i=13}^{14}(X - \tilde{\alpha}_i)) \in k_6[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,2}(B_{2,2}(B_{3,2}(B_{4,2}(B_{5,2,1}(\tilde{\alpha}_i)))))) \text{ for } 13 \leq i \leq 14$$

and store them. In addition, we factorize and upgrade $g^{(5)}_{2,2}$ *so as to be over* $k_6[X]$*. It is hold that* $g^{(5)}_{2,2}$ *is not irreducible, and we get that*

$$g^{(5)}_{2,2} = \prod_{i=15}^{16}(X - \tilde{\alpha}_i)) \in k_6[X].$$

We determine the roots of f *which are*

$$\alpha_i = B_0(B_{1,2}(B_{2,2}(B_{3,2}(B_{4,2}(B_{5,2,2}(\tilde{\alpha}_i)))))) \text{ for } 15 \leq i \leq 16$$

and store them.
The polynomial f *factors completely, which completes its splitting field computation.*

For the sake of completeness we list a table of examples in which we compare the two approaches -“Combined Approach”, (Ver2), and “Improved Approach”, (Ver3),- for the splitting field computation. The table can be found in Section A.2.

5.6 Reducible Polynomials

In this section we describe our approach of dealing with reducible polynomials. The aim is to generalize our already described approach to those polynomials as well.

Let $f \in K[X]$, where $K := \mathbb{F}_q((t))$ be a reducible polynomial. By factorizing we get that

$$f := f_1 \cdots f_l \in K[X].$$

For every irreducible factor f_i of f, we determine its corresponding field T_i for $1 \leq i \leq l$. Then, by using the same tools as before, we compute the composition T of T_i's with $1 \leq i \leq l$. That is to say,

$$T := T_1 \cdots T_l.$$

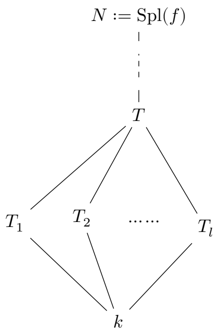


Figure 5.4: **Reducible Polynomials**

After the determination of T, we choose one of the irreducible factors of f and proceed with the same way as in the one described in our approach for irreducible polynomials.

Let us now analyze further our approach for reducible polynomials. We can compute the field T for Eisenstein polynomials. Thus, we reformulate our strategy of computing the field T_i for only the Eisenstein factors of f. To be more precise, after the factorization of f, we distinguish the irreducible factors into two lists of Eisenstein factors and non-Eisenstein, respectively. For the Eisenstein factors we compute their corresponding T_i's, and then we determine their composition T. We will thoroughly look into it below.

Having computed the field T, we consider the irreducible factors -Eisenstein and non-Eisenstein- over the field T, and start the usual procedure. In particular, we choose one of the Eisenstein factors of maximum degree so as to start the the steps of the splitting field computation. We note that we have stored the rest of irreducible factors -whether Eisenstein or not- in one list and we take it into account during execution of the algorithm. The computation of the splitting field is over as soon as all of the irreducible factors of f split completely.

We proceed now to describe the steps computing the field T. As we mentioned above, for every Eisenstein factor of f, we compute the corresponding field T_i. We create a list called "data" in which we append T_i together with some extra information provided that it is not already included.

Having done the above shieving, we distinguish two cases. If we have only one T_i provided by irreducible factors, then the composition T of T_i's would be simply $T := T_i$. Otherwise, we compute the composition of them by using the theory described after the Algorithm 6. In particular, this is done by employing our function "composite_Ti(Tis, U)" given that we have determined the new unramified extension U taking into account all T_i's. We note that the aforementioned function based on ideas and theory studied after the Algorithm 6. This way, we have computed the field T as an extension field. So then, we convert the field T to a local function field of the form $\mathbb{F}_{\tilde{q}}((u_{i_0}))$, which competes its computation.

It is worth pointing out that all the above operations for computing the composition T have been implemented in separate functions.

Chapter 6

Computation of Galois Groups over Local Function Fields

A Galois group is a group of field automorphisms under composition. Let $f \in K[X]$, where $K := \mathbb{F}_q((t))$, be a separable polynomial of degree n with splitting field $N := \mathrm{Spl}(f)$. The Galois group of f over K is defined to be the Galois group of the splitting field of f over K. Especially, $\mathrm{Gal}(\mathrm{Spl}(f)/K)$ is the group of the K-automorphisms of $\mathrm{Spl}(f)$, i.e. the automorphisms of $\mathrm{Spl}(f)$ which restricted to K are the identity. Moreover, if f is a separable polynomial, then $\mathrm{Gal}(\mathrm{Spl}(f)/K) \leqslant S_n$ acting on the roots $\alpha_1, \dots, \alpha_n$ of f. Thus, Galois groups are subgroups of symmetric groups.

We can speak about Galois groups of both irreducible or reducible polynomials. However, if f is an irreducible polynomial, then the Galois group of f is a transitive subgroup of the symmetric group S_n. A subgroup G of S_n is called transitive if for any $i \neq j$, with $i, j \in \{1, 2, \dots, n\}$, there is a permutation in G which sends i to j. Thus, given any two roots of f there is an automorphism in the Galois group $\mathrm{Gal}(\mathrm{Spl}(f)/K)$ of f over K which maps the first root to the second one.

6.1 Automorphisms as Permutations

Once we have defined the splitting field of a polynomial, we can ask what K-automorphisms exist on this field. We can view any automorphism on the splitting field as a permutation of the roots of the polynomial, which can be viewed as a permutation on the subscripts $\{1, 2, \dots, n\}$.

In our case, the splitting field of f is a field of formal Laurent series of the form $\mathbb{F}_{\tilde{q}}((u_\ell))$, where $\ell \geq 1$, that is $\mathrm{Spl}(f) = \mathbb{F}_{\tilde{q}}((u_\ell))$. The index ℓ in the variable of the splitting field indicates the number of steps needed for the splitting field computation, (see Figure (3.1) in Section 3.2).

To shorten notation, we write $N = \mathrm{Spl}(f) = \mathbb{F}_{\tilde{q}}((u))$, where $\tilde{q} = p^m$. We intend to construct the automorphisms of N over the base field $K := \mathbb{F}_q((t))$, where $q = p^l$. Once we have defined the splitting field of the polynomial, we get access to its roots. In this case, the roots of the polynomial are expressible only in terms of the variable of the splitting field. Also, in most cases the variable u of the splitting field is not a root of the original polynomial. Thus, we can define an automorphism σ of $\mathrm{Spl}(f)$ by determining $\sigma(u)$, in the case $\mathbb{F}_{\tilde{q}} = \mathbb{F}_q$. In the case of constant field extension, which means $\mathbb{F}_{\tilde{q}} \neq \mathbb{F}_q$, we can define an automorphism σ of $\mathrm{Spl}(f)$ if we determine where this automorphism sends u and the generating element w of $\mathbb{F}_{\tilde{q}}$ over $\mathbb{F}_q$.

Proposition 6.1.1
Let $N := \mathrm{Spl}(f) = \mathbb{F}_{\tilde{q}}((u))$, $\mathbb{F}_{\tilde{q}}^\times = \langle w \rangle$ and $\varphi \in \mathrm{Aut}(\mathbb{F}_{\tilde{q}})$. We define the map σ

$$\sigma: \quad N \to N, \quad \text{via } \sigma(u) = \sum_{i \geq 1} d_i u^i \text{ and } \sigma(w) = \varphi(w),$$

where $d_1 \neq 0$, $d_i \in \mathbb{F}_{\tilde{q}}$. Then it holds that the map σ is an injective homomorphism.

Proof. For this purpose, we should prove that for every $E_1, E_2 \in N$ we have that

$$\sigma(E_1 + E_2) = \sigma(E_1) + \sigma(E_2) \quad \text{and } \sigma(E_1 E_2) = \sigma(E_1)\sigma(E_2).$$

Since $d_1 \neq 0$, we have that $\sigma(u) = \sum\limits_{i\geq 1} d_i u^i = \varepsilon u$, where $\varepsilon \in \mathcal{O}_N^{\times}$.

Let assume that $E_1 = \sum\limits_{i\geq 0} \tilde{a}_i u^i$ and $E_2 = \sum\limits_{i\geq 0} \tilde{b}_i u^i$, with $\tilde{a}_i, \tilde{b}_i \in \mathbb{F}_{\tilde{q}}$. Then, $\sigma(E_1) = \sum\limits_{i\geq 0} \sigma(\tilde{a}_i)\sigma(u)^i = \sum\limits_{i\geq 0} \varphi(\tilde{a}_i)\varepsilon^i u^i$, and similarly $\sigma(E_2) = \sum\limits_{i\geq 0} \sigma(\tilde{b}_i)\sigma(u)^i = \sum\limits_{i\geq 0} \varphi(\tilde{b}_i)\varepsilon^i u^i$. So,

$$\begin{aligned}\sigma(E_1) + \sigma(E_2) &= \sum_{i\geq 0} \varphi(\tilde{a}_i)\varepsilon^i u^i + \sum_{i\geq 0} \varphi(\tilde{b}_i)\varepsilon^i u^i \\ &= \sum_{i\geq 0} (\varphi(\tilde{a}_i) + \varphi(\tilde{b}_i))\varepsilon^i u^i \overset{\varphi\in\mathrm{Aut}(\mathbb{F}_{\tilde{q}})}{=} \sum_{i\geq 0} \varphi(\tilde{a}_i + \tilde{b}_i)\sigma(u)^i = \sigma(E_1 + E_2).\end{aligned}$$

Moreover, we know that $E_1E_2 = \sum\limits_{\lambda\geq 0} \gamma_\lambda u^\lambda$, where $\gamma_\lambda = \sum\limits_{i+j=\lambda} \tilde{a}_i\tilde{b}_j = \sum\limits_{l=0}^{\lambda} \tilde{a}_{\lambda-l}\tilde{b}_l \in \mathbb{F}_{\tilde{q}}$. Then, $\sigma(E_1E_2) = \sum\limits_{\lambda\geq 0} \varphi(\gamma_\lambda)\varepsilon^\lambda u^\lambda$, since $\gamma_\lambda \in \mathbb{F}_{\tilde{q}}$, and

$$\sigma(E_1)\sigma(E_2) = (\sum_{i\geq 0} \varphi(\tilde{a}_i)\varepsilon^i u^i)(\sum_{j\geq 0} \varphi(\tilde{b}_i)\varepsilon^i u^i) = \sum_{\lambda\geq 0} \tilde{\gamma}_\lambda \varepsilon^\lambda u^\lambda,$$

where $\tilde{\gamma}_\lambda = \sum\limits_{i+j=\lambda} \varphi(\tilde{a}_i)\varphi(\tilde{b}_j) = \sum\limits_{l=0}^{\lambda} \varphi(\tilde{a}_{\lambda-l})\varphi(\tilde{b}_l) \overset{\varphi\in\mathrm{Aut}(\mathbb{F}_{\tilde{q}})}{=} \varphi(\sum\limits_{l=0}^{\lambda} \tilde{a}_{\lambda-l}\tilde{b}_l) = \varphi(\gamma_\lambda)$, which completes the proof. Thus, σ is a homomorphism, and so σ is an injective homomorphism as N is a field. □

Lemma 6.1.2
Let $f \in K[X]$, where $K = \mathbb{F}_q((t))$ and σ an injective homomorphism as defined in Proposition 6.1.1. Also, assume that $\sigma|_K = \mathrm{id}$. Then, σ is a K-automorphism of N, i.e $\sigma \in \mathrm{Aut}(N/K)$, which means that σ induces a permutation of the roots of f.

Proof. By hypothesis, we know that the map σ is an injective homomorphism and $\sigma|_K = \mathrm{id}$, and thus σ is an automorphism of N, since N/K is a finite extension (see Theorem 2.8.2, [40]). Therefore, σ is a K-automorphism of N. □

Let us analyze the automorphism in the case of a constant field extension, which is nothing else but the Frobenius automorphism.

Remark 6.1.3
Let $L_\ell = \mathrm{Spl}(f)$ and $L_\ell/L_{\ell-1}$ be a constant field extension. Then $\sigma \in \mathrm{Aut}(L_\ell/K)$ is determined as follows

$$\sigma : L_\ell \longrightarrow L_\ell, \textit{ with } w \mapsto w^{q_{\ell-1}} \textit{ and } u_\ell \mapsto u_\ell.$$

If the constant field extension occurs at an intermediate step, then we extend the Frobenius automorphism to get $\sigma \in \mathrm{Aut}(L_\ell/K)$, which is analyzed in the remark below.

Remark 6.1.4
Let L_i/L_{i-1} be a constant field extension. Then we determine $\sigma \in \mathrm{Aut}(L_\ell/L_{i-1})$ as follows

$$\sigma : L_\ell \longrightarrow L_\ell, \textit{ with } w \mapsto w^{q_{i-1}} \textit{ and } u_\ell \mapsto \sigma(u_\ell) \textit{ such that } \sigma(\mathcal{E}(u_\ell)) = \mathcal{E}(u_\ell)$$

where $\mathcal{E}(u_\ell) := \Phi_\ell \circ \cdots \circ \Phi_{i+1}(u_i) \in L_\ell$.

Proof. Since the constant field extension occurs at step i, we can construct the following Frobenius automorphism

$$\sigma_{L_i} : L_i \longrightarrow L_i, \text{ with } w \mapsto w^{q_{i-1}} \text{ and } u_i \mapsto u_i.$$

We extend σ_{L_i} to an automorphism σ over L_ℓ, i.e. $\sigma|_{L_i} = \sigma_{L_i}$, such that $\sigma(w) = w^{q_{i-1}}$. To do so, it suffices to find $\sigma(u_\ell)$. By using the relations $\Phi_1(t) \in L_1, \Phi_{j+1}(u_j) \in L_{j+1}$, with $1 \leq j \leq \ell - 1$ of the variables, (see Figure 3.1 too), we can express u_i as an element in L_ℓ as follows

$$\Phi_\ell \circ \cdots \circ \Phi_{i+1}(u_i) = \sum_{\kappa\geq 1} \delta_\kappa u_\ell^\kappa =: \mathcal{E}(u_\ell) \in L_\ell. \tag{6.1}$$

Combining the properties $\sigma_{L_i}(u_i) = u_i$ and $\sigma|_{L_i} = \sigma_{L_i}$, it yields that

$$\sigma(\mathcal{E}(u_\ell)) = \mathcal{E}(u_\ell). \tag{6.2}$$

□

Remark 6.1.5
Solving the equation (6.2) results in finding that $\sigma(u_\ell) = \sum\limits_{\kappa\geq 1} d_\kappa u_\ell^\kappa$ with $d_\kappa \in \mathbb{F}_{q_\ell}$. For the approach solving such an equation (6.2), we will thoroughly look into it below.

Remark 6.1.6
It is worth noting that the number of σ as defined in Remark 6.1.4 is equal to $[L_\ell : L_i]$ if the constant field extension occurs at step i.

For the general case, i.e. non constant field extensions, we determine $\sigma(u_\ell)$ by solving the equation $\sigma(\alpha_1) = \alpha_2$. For that case, we completely define an automorphism σ of $\mathrm{Spl}(f) = L_\ell$ by the determination of $\sigma(u_\ell)$, since $\sigma(w) = w$. For simplicity of the notation we define $u_\ell := u$ and $q_\ell := \tilde{q}$.

Let σ be an automorphism of $\mathrm{Spl}(f) = \mathbb{F}_{\tilde{q}}((u))$ which maps the root α_1 to α_2. We would like to determine the image of u such that $\sigma(\alpha_1) = \alpha_2$. We suppose that

$$\sigma(u) = \sum_{i\geq 1} d_i u^i \quad \text{with } d_i \in \mathbb{F}_{\tilde{q}}$$

and we set

$$S_r(u) := \sum_{i=1}^{r} d_i u^i \in \mathbb{F}_{\tilde{q}}[u].$$

Let now

$$\alpha_1 = \alpha_1(u) = \sum_{i\geq \mathrm{val}} a_i u^i, \text{ and } \alpha_2 = \alpha_2(u) = \sum_{i\geq \mathrm{val}} b_i u^i, \tag{6.3}$$

where val denotes the valuation of the roots of the polynomial, i.e $\mathrm{val} := v_u(\alpha)$, and $a_{\mathrm{val}} \neq 0$, $b_{\mathrm{val}} \neq 0$, then we have that

$$\alpha_1(S_r(u)) = \sum_{i\geq \mathrm{val}} a_i (S_r(u))^i,$$

$$\sigma(\alpha_1(u)) = \sum_{i\geq \mathrm{val}} a_i \sigma(u)^i = \alpha_1(\sigma(u))$$

and let

$$\alpha_1(\sigma(u)) = \sum_{i\geq \mathrm{val}} e_i u^i.$$

Under the conditions stated above and by Proposition 6.1.1 and Lemma 6.1.2, we know that $\sigma \in \mathrm{Aut}(N/K)$. In what follows we study how we determine $\sigma(u)$ by solving the equation $\sigma(\alpha_1) = \alpha_2$. Then, we can prove the following lemma.

Lemma 6.1.7
For $\mathrm{val} \geq 0$ *and* $r \geq 1$ *we get*

$$\alpha_1(\sigma(u)) \equiv \alpha_1(S_r(u)) \mod u^\kappa, \text{ where } \kappa := r + \mathrm{val}.$$

Proof. We prove it by using induction. Firstly, for $r = 1$ we have that $S_1(u) = d_1 u$ and then

$$\alpha_1(S_1(u)) = \sum_{i\geq \mathrm{val}} a_i (d_1 u)^i \equiv a_{\mathrm{val}}\, d_1^{\mathrm{val}}\, u^{\mathrm{val}} \equiv \sum_{i\geq \mathrm{val}} a_i \sigma(u)^i = \alpha_1(\sigma(u)) \mod u^{1+\mathrm{val}}.$$

We suppose that the statement holds for r. This means that $\alpha_1(\sigma(u)) \equiv \alpha_1(S_r(u)) \mod u^{r+\mathrm{val}}$. Then we prove it for $r+1$, that is

$$\alpha_1(\sigma(u)) \equiv \alpha_1(S_{r+1}(u)) \mod u^{r+1+\mathrm{val}}.$$

We have that $S_{r+1}(u) = S_r(u) + d_{r+1}u^{r+1}$, and so then

$$\begin{aligned}
\alpha_1(S_{r+1}(u)) &= \alpha_1(S_r(u) + d_{r+1}u^{r+1}) = \sum_{i\geq \mathrm{val}} a_i (S_r(u) + d_{r+1}u^{r+1})^i \\
&\equiv \sum_{i=\mathrm{val}}^{r+\mathrm{val}} a_i (S_r(u) + d_{r+1}u^{r+1})^i \mod u^{r+1+\mathrm{val}} \\
&\equiv \sum_{i=\mathrm{val}}^{r+\mathrm{val}} a_i (S_r^i(u) + iS_r^{i-1}(u)d_{r+1}u^{r+1} + \frac{i(i-1)}{2} S_r^{i-2}(u) d_{r+1}^2 (u^{r+1})^2 + \cdots \\
&\qquad + iS_r(u)d_{r+1}^{i-1}(u^{r+1})^{i-1} + d_{r+1}^i (u^{r+1})^i) \mod u^{r+1+\mathrm{val}}.
\end{aligned}$$

Since $u^{i-1} \mid S_r^{i-1}(u)$, we have that

$$\sum_{i=\mathrm{val}}^{r+\mathrm{val}} a_i i S_r^{i-1}(u) d_{r+1} u^{r+1} \equiv \mathrm{val}\, a_{\mathrm{val}}\, S_r^{\mathrm{val}-1}(u)\, d_{r+1} u^{r+1} \mod u^{r+1+\mathrm{val}},$$

as for $i \geq \mathrm{val}+1$ we have that $i - 1 + r + 1 \geq r + 1 + \mathrm{val}$.

Also, $\sum_{i=\mathrm{val}}^{r+\mathrm{val}} (a_i \frac{i(i-1)}{2} S_r^{i-2}(u) d_{r+1}^2 (u^{r+1})^2) \equiv 0 \mod u^{r+1+\mathrm{val}}$, since $r + \mathrm{val} < 2r + \mathrm{val} \leq 2r + i \leq$

$3r+\mathrm{val}$. Similarly, we can prove that all the rest terms are 0 modulo $u^{r+1+\mathrm{val}}$. Then we have that

$$\begin{aligned}\alpha_1(S_{r+1}(u)) &\equiv \sum_{i=\mathrm{val}}^{r+\mathrm{val}} a_i(S_r^i(u)+iS_r^{i-1}(u)d_{r+1}u^{r+1}) \\ &\equiv \left(\sum_{i=\mathrm{val}}^{r+\mathrm{val}} a_i S_r^i(u)\right) + \mathrm{val}\, a_{\mathrm{val}}\, S_r^{\mathrm{val}-1}(u)\, d_{r+1} u^{r+1} \\ &\equiv \alpha_1(S_r(u)) + \mathrm{val}\, a_{\mathrm{val}}\, d_1^{\mathrm{val}-1}\, d_{r+1}\, u^{r+\mathrm{val}} \mod u^{r+1+\mathrm{val}}.\end{aligned}$$

Hence, for $\mathrm{val} \geq 0$ we get that

$$\alpha_1(S_{r+1}(u)) \equiv \alpha_1(S_r(u)) + \mathrm{val}\, a_{\mathrm{val}}\, d_1^{\mathrm{val}-1}\, d_{r+1}\, u^{r+\mathrm{val}} \mod u^{r+1+\mathrm{val}}.$$

By induction hypothesis, we know that $\alpha_1(S_r(u)) \equiv \alpha_1(\sigma(u)) \mod u^{r+\mathrm{val}}$, which means that

$$\alpha_1(S_r(u)) = \left(\sum_{i=\mathrm{val}}^{r+\mathrm{val}-1} e_i u^i\right) + \tilde{f}_{r+\mathrm{val}}\, u^{r+\mathrm{val}}, \quad \text{where} \quad \tilde{f}_{r+\mathrm{val}} \in \mathbb{F}_q((u))$$

Thus, the induction hypothesis yields that

$$\alpha_1(S_{r+1}(u)) \equiv \left(\sum_{i=\mathrm{val}}^{r+\mathrm{val}-1} e_i u^i\right) + \tilde{e}_{r+\mathrm{val}}\, u^{r+\mathrm{val}} + \mathrm{val}\, a_{\mathrm{val}}\, d_1^{\mathrm{val}-1}\, d_{r+1} u^{r+\mathrm{val}} \mod u^{r+1+\mathrm{val}}.$$

The element $\tilde{e}_{r+\mathrm{val}} \in \mathbb{F}_q[d_1, d_2, \dots, d_r]$ is the coefficient of the term $u^{r+\mathrm{val}}$ arising from the term $\tilde{f}_{r+\mathrm{val}}\, u^{r+\mathrm{val}}$. Therefore, we have that

$$\alpha_1(S_{r+1}(u)) \equiv \sum_{i=\mathrm{val}}^{r+\mathrm{val}} e_i u^i \equiv \alpha_1(\sigma(u)) \mod u^{r+1+\mathrm{val}},$$

since $\tilde{e}_{r+\mathrm{val}} u^{r+\mathrm{val}} + \mathrm{val} a_{\mathrm{val}} d_1^{\mathrm{val}-1} d_{r+1} u^{r+\mathrm{val}} \equiv e_{r+\mathrm{val}} u^{r+\mathrm{val}} \mod u^{r+1+\mathrm{val}}$. □

Thus, according to Lemma 6.1.7 the equation $\sigma(\alpha_1) = \alpha_2$ implies

$$\alpha_1(S_r(u)) \equiv \alpha_2(u) \mod u^{r+\mathrm{val}}.$$

This means that it is sufficient to determine a "partial" image $S_r(u)$ of u such that $\sigma(\alpha_1) = \alpha_2$. This allows us to compute the image of u by computing successively the coefficients $\{d_i\}_{i=1}^r$.

Proposition 6.1.8
Let $\mathrm{Spl}(f) = \mathbb{F}_q((u))$ and $\sigma \in \mathrm{Aut}(N/K)$ such that $\sigma(\alpha_1) = \alpha_2$. Let also $\mathrm{val} = v_u(\alpha_1) = v_u(\alpha_2)$ be the u-valuations of α_1 and α_2 such that $\mathrm{val} \geq 1$ and $p \nmid \mathrm{val}$, where $p = \mathrm{Char}(\mathbb{F}_q)$. Denote the image of u under σ by

$$\sigma(u) = \sum_{i \geq 1} d_i u^i.$$

Then, d_1 is determined by the equation

$$a_{\mathrm{val}}\, d_1^{\mathrm{val}} = b_{\mathrm{val}}$$

and for every $r \geq 1$ the d_{r+1} is determined by the equation

$$\tilde{e}_{r+\mathrm{val}} + \mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1} d_{r+1} = b_{r+\mathrm{val}},$$

where $\tilde{e}_{r+\mathrm{val}} \in \mathbb{F}_q[d_1, \dots, d_r]$.

Proof. By Lemma 6.1.7, the equation $\sigma(\alpha_1) = \alpha_2$ implies

$$\alpha_1(S_r(u)) \equiv \alpha_2(u) \mod u^{r+\mathrm{val}}. \tag{6.4}$$

The equation (6.4) for $r = 1$ gives that

$$\alpha_1(S_1(u)) \equiv \alpha_2(u) \mod u^{1+\mathrm{val}} \Leftrightarrow a_{\mathrm{val}}\, d_1^{\mathrm{val}}\, u^{\mathrm{val}} \equiv b_{\mathrm{val}}\, u^{\mathrm{val}} \mod u^{1+\mathrm{val}}$$

and equivalently we obtain the following equation of d_1

$$a_{\mathrm{val}}\, d_1^{\mathrm{val}} = b_{\mathrm{val}}. \tag{6.5}$$

In addition, according to the proof of Lemma 6.1.7, for every $r \geq 1$ we have the following equation

$$\alpha_1(S_{r+1}(u)) \equiv (\sum_{i=\mathrm{val}}^{r+\mathrm{val}-1} e_i u^i) + \tilde{e}_{r+\mathrm{val}}\, u^{r+\mathrm{val}} + \mathrm{val}\, a_{\mathrm{val}}\, d_1^{\mathrm{val}-1}\, d_{r+1}\, u^{r+\mathrm{val}} \mod u^{r+1+\mathrm{val}}$$

and the element $\tilde{e}_{r+\mathrm{val}} \in \mathbb{F}_q[d_1, \dots, d_r]$ is the coefficient of the term $u^{r+\mathrm{val}}$ arising from the induction hypothesis, (see proof of Lemma 6.1.7). Then for every $r \geq 1$ we have that

$$\alpha_1(S_{r+1}(u)) \equiv \alpha_2(u) \mod u^{r+1+\mathrm{val}} \Rightarrow$$
$$\left(\sum_{i=\mathrm{val}}^{r-1+\mathrm{val}} e_i u^i\right) + \tilde{e}_{r+\mathrm{val}}\, u^{r+\mathrm{val}} + \mathrm{val}\, a_{\mathrm{val}}\, d_1^{\mathrm{val}-1}\, d_{r+1}\, u^{r+\mathrm{val}} \equiv \sum_{i=\mathrm{val}}^{r+\mathrm{val}} b_i u^i \mod u^{r+1+\mathrm{val}}.$$

This yields that
$$\tilde{e}_{r+\mathrm{val}} + \mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1} d_{r+1} = b_{r+\mathrm{val}}. \tag{6.6}$$
□

According to Proposition 6.1.8, we have that the equation of d_1, which is given by the equation (6.5), is of degree equal to val. In the case $p \nmid \mathrm{val}$ and $\mathrm{val} \geq 1$ the equation of d_{r+1}, which is given by the equation (6.6), is linear for each $r \geq 1$. In particular, once we have chosen the value of d_1, the value of d_{r+1} is uniquely determined for every $r \geq 1$.

Corollary 6.1.9
Let $p \nmid \mathrm{val}$ and $\mathrm{val} \neq 0$. Then, for each $r \geq 1$ we have that
$$d_{r+1} = \frac{b_{r+\mathrm{val}} - \tilde{e}_{r+\mathrm{val}}}{\mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1}}, \tag{6.7}$$
where $d_{r+1} \in \mathbb{F}_{\tilde{q}}$ and $a_{\mathrm{val}}\, d_1 \neq 0$.

It is clear that the denominator of the formula (6.7) is divisible by 0 in the case $p \mid \mathrm{val}$. However, in the case $p \mid \mathrm{val}$ we know that $p \nmid \mathrm{val} + 1$, so then we repeat the above computations and we will determine the value of d_{r+1} by the equation
$$\mathfrak{F} := \mathrm{Expression}(a_1, \dots, a_j, d_1, \dots, d_{j-\mathrm{val}+1}) = b_j,$$
for some $j > r + \mathrm{val}$. Consequently, in this case we do not know the degree of the equation of d_{r+1}, with $r \geq 1$. We will analyze this case later.

For the sake of clarity, we give the equations described in Proposition 6.1.8 in the case $\mathrm{val} = 1$.

Remark 6.1.10
For simplicity we suppose that $\mathrm{val} = 1$ and we compute for every $r \geq 1$ the equation of d_r by using the equation
$$\alpha_1(S_r(u)) \equiv \alpha_2(u) \mod u^{r+\mathrm{val}}.$$
Then, we successively obtain the following system of equations
$$\begin{aligned}
a_1 d_1 &= b_1 && (6.8)\\
a_2 d_1^2 + a_1 d_2 &= b_2 && (6.9)\\
a_1 d_3 + 2a_2 d_1 d_2 + a_3 d_1^3 &= b_3 && (6.10)\\
a_1 d_4 + a_2 d_2^2 + 2a_2 d_1 d_3 + 3a_3 d_1^2 d_2 + a_4 d_1^4 &= b_4 && (6.11)\\
a_1 d_5 + 2a_2 d_2 d_3 + 2a_2 d_1 d_4 + 3a_3 d_1 d_2^2 + 3a_3 d_1^2 d_3 + 4a_4 d_1^3 d_2 + a_5 d_1^5 &= b_5 && (6.12)\\
a_1 d_6 + 2a_2 d_1 d_5 + 2a_2 d_2 d_4 + a_2 d_3^2 + 3a_3 d_1^2 d_4 + 6a_3 d_1 d_2 d_3 + a_3 d_2^3 + & \\
+4a_4 d_1^3 d_3 + 6a_4 d_1^2 d_2^2 + 5a_5 d_1^4 d_2 + a_6 d_1^6 &= b_6 && (6.13)\\
&\ \vdots
\end{aligned}$$

For the purpose of computing $\sigma(u)$, we should construct the equation of each coefficient d_i of $\sigma(u)$. As we have mentioned above in the case $p \nmid \mathrm{val}$ and $\mathrm{val} \neq 0$ we have an equation of d_i, for each $i \geq 1$. However, the problem is that we do not know the value of the corresponding $\tilde{e}_{r+\mathrm{val}}$ in the equation (6.7) for every $r \geq 1$. Therefore, we would like to determine the value of $\tilde{e}_{r+\mathrm{val}}$.

We notice a nice relation between the partitions of the number m and the equation of $d_{m-\mathrm{val}+1}$,
$$\tilde{e}_m + \mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1} d_{m-\mathrm{val}+1} = b_m.$$
We denote $\mathfrak{F} := \tilde{e}_m + \mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1} d_{m-\mathrm{val}+1}$ and $\tilde{e}_m \in \mathbb{F}_q[d_1, \dots, d_{m-\mathrm{val}}]$, especially
$$\tilde{e}_m = \sum_{\substack{\forall s \leq m \\ \forall e_1 + \cdots + e_{j_0} = s}} w_s a_s \prod_{i=1}^{j_0} d_i^{e_i}, \tag{6.14}$$

with $w_s \in \mathbb{F}_q$, for every $s \leq m$. The sum is taken over all combinations of non-negative integers indices e_1 through e_{j_0} such that $e_1 + \cdots + e_{j_0} = s$ with $j_0 \leq m - \mathrm{val}$, for every $s \leq m$. This relation enables us to construct the equation of $d_{m-\mathrm{val}+1}$ by using the partitions of m.

We would like to recall the following definition for multinomial coefficients.

Definition 6.1.11
Let $s = \sum_{i=1}^{\bar{r}} e_i$, then the multinomial coefficient is defined via

$$\binom{s}{e_1, \dots, e_{\bar{r}}} = \binom{s}{e_1}\binom{s-e_1}{e_2}\binom{s-(e_1+e_2)}{e_3} \cdots \binom{e_{\bar{r}-1}+e_{\bar{r}}}{e_{\bar{r}-1}}\binom{e_{\bar{r}}}{e_{\bar{r}}} = \frac{s!}{e_1! e_2! \cdots e_{\bar{r}}!}.$$

Before presenting the aforementioned relation, we explain the way that each partition of a number m corresponds to one term of $\mathfrak{F}$.

Remark 6.1.12
Each partition $\mathfrak{p} := \underbrace{c_1 + \cdots + c_1}_{e_1} + \underbrace{c_2 + \cdots + c_2}_{e_2} + \ldots + \underbrace{c_{\bar{r}} + \cdots + c_{\bar{r}}}_{e_{\bar{r}}}$ of the number $m := \sum_{i=1}^{\bar{r}} c_i e_i$ corresponds to a term of the form

$$\mathfrak{z}_{\mathfrak{p}} := w_{\mathfrak{p}} a_s \prod_{i=1}^{\bar{r}} d_{c_i}^{e_i},$$

where $s = \sum_{i=1}^{\bar{r}} e_i$ is the number of summands of the partition $\mathfrak{p}$ and the coefficient $w_{\mathfrak{p}} \in \mathbb{F}_q$ is defined as

$$w_{\mathfrak{p}} := \binom{s}{e_1, \dots, e_{\bar{r}}} = \frac{s!}{e_1! e_2! \cdots e_{\bar{r}}!}.$$

Comment 6.1.13
It is worth pointing out that $c_i \neq c_j$ for every $i \neq j$ with respect to our notation in Remark 6.1.12.

Example 6.1.14
Let $m = 7$ and $\mathfrak{p} := 3 + 2 + 1 + 1$. Then, $m = 3 + 2 + 2 \cdot 1$ and the corresponding term is

$$\mathfrak{z}_{\mathfrak{p}} := w_{\mathfrak{p}} a_4 d_3 d_2 d_1^2,$$

where $w_{\mathfrak{p}} = \frac{4!}{1!1!2!} = 12$.

Proposition 6.1.15
Let $\mathfrak{F} = \mathfrak{F}_{d_{m-\mathrm{val}+1}} := \tilde{e}_m + \mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1} d_{m-\mathrm{val}+1}$ and

$$\mathfrak{F} = b_m$$

be the equation of $d_{m-\mathrm{val}+1}$ in case σ maps α_1 to α_2. Let also $\{\mathfrak{p}_1, \dots, \mathfrak{p}_{p(m)}\}$ be the possible partitions of m, where the partition function $p(m)$ is the number of possible partitions of m. Then,

$$\mathfrak{F} = \sum_{i=1}^{p(m)} \mathfrak{z}_{\mathfrak{p}_i},$$

where $\mathfrak{z}_{\mathfrak{p}_i}$ is the corresponding term of the partition $\mathfrak{p}_i$ as defined in Remark 6.1.12.

Proof. By hypothesis, we have that $\mathfrak{F} = \mathfrak{F}_{d_{m-\mathrm{val}+1}} := \tilde{e}_m + \mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1} d_{m-\mathrm{val}+1}$ and the term $\tilde{e}_m$ is given by (6.14). This term can be rewritten as follows

$$\tilde{e}_m = \sum_{\substack{\forall s \leq m \\ \forall E = [e_1, \dots, e_{j_0}] \text{ s.t.} \\ e_1 + \cdots + e_{j_0} = s \\ \forall j_0 \leq m - \mathrm{val}}} w_{E,s} a_s \prod_{i=1}^{j_0} d_{c_i}^{e_i}, \tag{6.15}$$

where $m := \sum_{i=1}^{j_0} c_i e_i$, $s = \sum_{i=1}^{j_0} e_i$ and $w_{E,s} := \binom{s}{e_1, \dots, e_{\bar{r}}} = \frac{s!}{e_1! e_2! \cdots e_{\bar{r}}!}$. Taking also into account the term $\mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1} d_{m-\mathrm{val}+1}$ which corresponds to the case $[c_1, c_2] = [1, m - \mathrm{val} + 1]$, $[e_1, e_2] = [\mathrm{val} - 1, 1]$, $s = \mathrm{val}$ and $w_{E,s} = \mathrm{val}$, we can conclude $\mathfrak{F} := \sum_{j=1}^{p(m)} \mathfrak{z}_{\mathfrak{p}_j}$, where $\mathfrak{z}_{\mathfrak{p}_j}$ as defined in Remark 6.1.12. □

We thoroughly examine the aforementioned relation by studying an example, which is given in the remark below.

Remark 6.1.16

For simplicity, we suppose that val $= 2$ *which means that* $a_1 = 0 = b_1$ *and we compute for every* $r \geq 1$ *the equation of* d_r *by the equation*

$$\alpha_1(S_r(u)) \equiv \alpha_2(u) \mod u^{r+val}$$

and we successively obtain the following system of equations

$$\begin{aligned}
a_2d_1^2 &= b_2 & (6.16)\\
2a_2d_1d_2 + a_3d_1^3 &= b_3 & (6.17)\\
a_2d_2^2 + 2a_2d_1d_3 + 3a_3d_1^2d_2 + a_4d_1^4 &= b_4 & (6.18)\\
2a_2d_2d_3 + 2a_2d_1d_4 + 3a_3d_1d_2^2 + 3a_3d_1^2d_3 + 4a_4d_1^3d_2 + a_5d_1^5 &= b_5 & (6.19)\\
2a_2d_1d_5 + 2a_2d_2d_4 + a_2d_3^2 + 3a_3d_1^2d_4 + 6a_3d_1d_2d_3 + a_3d_2^3 + &\\
+4a_4d_1^3d_3 + 6a_4d_1^2d_2^2 + 5a_5d_1^4d_2 + a_6d_1^6 &= b_6 & (6.20)\\
&\vdots
\end{aligned}$$

Now we describe the way to construct the above equations by using the partitions.

$\leadsto$ ***For*** d_1**:**

According to equation (6.16), we obtain the value of d_1. *Now let us check the partitions of* 2.

Partitions of 2:

$$\begin{array}{lllll}
2 = & 1+1 & \leftrightarrow & a_2d_1^2 & \textit{where} \;\; w = 1 = \binom{2}{2}\\
= & 2 & \leftrightarrow & a_1d_2 & \textit{where} \;\; w = 1 = \binom{1}{1}
\end{array}$$

So,

$$\mathfrak{F}_{d_1} = a_2d_1^2 + a_1d_2.$$

Thus, each term of the equation of d_1, *(in this case, equation (6.16)), corresponds to a partition of* 2, *since* $a_1 = 0$.

$\leadsto$ ***For*** d_2**:**

Partitions of 3:

$$\begin{array}{lllll}
3 = & 1+1+1 & \leftrightarrow & a_3d_1^3, & \textit{where} \;\; w = 1 = \binom{3}{3}\\
= & 1+2 & \leftrightarrow & 2a_2d_1d_2, & \textit{where} \;\; w = 2 = \binom{2}{1}\binom{1}{1}\\
= & 3 & \leftrightarrow & a_1d_3, & \textit{where} \;\; w = 1 = \binom{1}{1}
\end{array}$$

So,

$$\mathfrak{F}_{d_2} = a_1d_3 + 2a_2d_1d_2 + a_3d_1^3 = 2a_2d_1d_2 + a_3d_1^3,$$

since $a_1 = 0$, *and then the right side of the equation (6.17) is constructed by the partitions of* 3.

$\leadsto$ ***For*** d_3**:**

Partitions of 4:

$$\begin{array}{lllll}
4 = & 1+1+1+1 & \leftrightarrow & a_4d_1^4, & \textit{where} \;\; w = 1 = \binom{4}{4}\\
= & 1+1+2 & \leftrightarrow & 3a_3d_1^2d_2, & \textit{where} \;\; w = 3 = \binom{3}{2}\binom{1}{1}\\
= & 1+3 & \leftrightarrow & 2a_2d_1d_3, & \textit{where} \;\; w = 2 = \binom{2}{1}\binom{1}{1}\\
= & 2+2 & \leftrightarrow & a_2d_2^2, & \textit{where} \;\; w = 1 = \binom{2}{2}\\
= & 4 & \leftrightarrow & a_1d_4, & \textit{where} \;\; w = 1 = \binom{1}{1}
\end{array}$$

So,

$$\begin{aligned}
\mathfrak{F}_{d_3} &= a_1d_4 + a_2d_2^2 + 2a_2d_1d_3 + 3a_3d_1^2d_2 + a_4d_1^4\\
&= a_2d_2^2 + 2a_2d_1d_3 + 3a_3d_1^2d_2 + a_4d_1^4,
\end{aligned}$$

since $a_1 = 0$, *and then the right side of the equation (6.18) is constructed by the partitions of* 4.

$\leadsto$ ***For*** d_4**:**

Partitions of 5:

$$\begin{array}{rllcll}
5 = & 1+1+1+1+1 & \leftrightarrow & a_5 d_1^5, & \textit{where} & w = 1 = \binom{5}{5} \\
= & 1+1+1+2 & \leftrightarrow & 4a_4 d_1^3 d_2, & \textit{where} & w = 4 = \binom{4}{3}\binom{1}{1} \\
= & 1+1+3 & \leftrightarrow & 3a_3 d_1^2 d_3, & \textit{where} & w = 3 = \binom{3}{2}\binom{1}{1} \\
= & 1+2+2 & \leftrightarrow & 3a_3 d_1 d_2^2, & \textit{where} & w = 3 = \binom{3}{1}\binom{2}{2} \\
= & 1+4 & \leftrightarrow & 2a_2 d_1 d_4, & \textit{where} & w = 2 = \binom{2}{1}\binom{1}{1} \\
= & 2+3 & \leftrightarrow & 2a_2 d_2 d_3, & \textit{where} & w = 2 = \binom{2}{1}\binom{1}{1} \\
= & 5 & \leftrightarrow & a_1 d_5, & \textit{where} & w = 1 = \binom{1}{1}
\end{array}$$

Thus,

$$\begin{aligned}
\mathfrak{F}_{d_4} &= a_1 d_5 + 2a_2 d_2 d_3 + 2a_2 d_1 d_4 + 3a_3 d_1 d_2^2 + 3a_3 d_1^2 d_3 + 4a_4 d_1^3 d_2 + a_5 d_1^5 \\
&= 2a_2 d_2 d_3 + 2a_2 d_1 d_4 + 3a_3 d_1 d_2^2 + 3a_3 d_1^2 d_3 + 4a_4 d_1^3 d_2 + a_5 d_1^5,
\end{aligned}$$

since $a_1 = 0$, *and then the right side of the equation* (6.19) *is constructed by the partitions of* 5.

↝ ***For*** d_5***:***
Partitions of 6:

$$\begin{array}{rllcll}
6 = & 1+1+1+1+1+1 & \leftrightarrow & a_6 d_1^6, & \textit{where} & w = 1 = \binom{6}{6} \\
= & 1+1+1+1+2 & \leftrightarrow & 5a_5 d_1^4 d_2, & \textit{where} & w = 5 = \binom{5}{4}\binom{1}{1} \\
= & 1+1+2+2 & \leftrightarrow & 6a_4 d_1^2 d_2^2, & \textit{where} & w = 6 = \binom{4}{2}\binom{2}{2} \\
= & 2+2+2 & \leftrightarrow & a_3 d_2^3, & \textit{where} & w = 1 = \binom{3}{3} \\
= & 1+1+1+3 & \leftrightarrow & 4a_4 d_1^3 d_3, & \textit{where} & w = 4 = \binom{4}{3}\binom{1}{1} \\
= & 1+2+3 & \leftrightarrow & 6a_3 d_1 d_2 d_3, & \textit{where} & w = 6 = \binom{3}{1}\binom{2}{1}\binom{1}{1} \\
= & 3+3 & \leftrightarrow & a_2 d_3^2, & \textit{where} & w = 1 = \binom{2}{2} \\
= & 1+1+4 & \leftrightarrow & 3a_3 d_1^2 d_4, & \textit{where} & w = 3 = \binom{3}{2}\binom{1}{1} \\
= & 2+4 & \leftrightarrow & 2a_2 d_2 d_4, & \textit{where} & w = 2 = \binom{2}{1}\binom{1}{1} \\
= & 1+5 & \leftrightarrow & 2a_2 d_1 d_5, & \textit{where} & w = 2 = \binom{2}{1}\binom{1}{1} \\
= & 6 & \leftrightarrow & a_1 d_6, & \textit{where} & w = 1 = \binom{1}{1}
\end{array}$$

Hence,

$$\begin{aligned}
\mathfrak{F}_{d_5} &= a_1 d_6 + 2a_2 d_1 d_5 + 2a_2 d_2 d_4 + a_2 d_3^2 + 3a_3 d_1^2 d_4 + 6a_3 d_1 d_2 d_3 + a_3 d_2^3 + \\
&\quad + 4a_4 d_1^3 d_3 + 6a_4 d_1^2 d_2^2 + 5a_5 d_1^4 d_2 + a_6 d_1^6 \\
&= 2a_2 d_1 d_5 + 2a_2 d_2 d_4 + a_2 d_3^2 + 3a_3 d_1^2 d_4 + 6a_3 d_1 d_2 d_3 + a_3 d_2^3 + \\
&\quad + 4a_4 d_1^3 d_3 + 6a_4 d_1^2 d_2^2 + 5a_5 d_1^4 d_2 + a_6 d_1^6,
\end{aligned}$$

since $a_1 = 0$, *and then the right side of the equation* (6.20) *is constructed by the partitions of* 6.

Therefore, by the above computations, we are able to construct the equation of d_m *by using the partitions of the number* $m + 1$.

In general, the above computations are true for every val. *In the general case, we start with an equation of degree* val *for* d_1. *This means that the equation of* d_1 *is of the form*

$$a_{\mathrm{val}} d_1^{\mathrm{val}} = b_{\mathrm{val}},$$

since $a_1 = \cdots = a_{\mathrm{val}-1} = 0$, *and similarly we compute the equation of* d_i, *for every* $i \geq 2$ *by using the partitions of the number* $\mathrm{val} + i - 1$.

Therefore, we can construct the equation (6.6) of d_{r+1}, for every $r \geq 1$, by using the partitions of the number $r + \mathrm{val}$. Thus, we can determine the value of $\tilde{e}_{r+\mathrm{val}}$, and consequently the value of d_{r+1}.

This method depends on the number of possible partitions of the positive integer $m := r + \mathrm{val}$. We already know that the partition function $p(m)$ grows as an exponential function of the square root of its argument. This means that as the number m increases, then we have to handle with a big number $p(m)$ of partitions, and consequently a big number of terms, in order to construct the equation of $d_{r+1} = d_{m-\mathrm{val}+1}$. As a result, checking all of those possibilities is time-consuming. However, we notice that many of those possibilities do not generate a non-zero term of the equation. Thus, we should apply some sieving techniques for the purpose of keeping only the useful ones.

According to Remark 6.1.12, each term consist of three components, which are w, a_s and the product $\prod_{i=1}^{\bar{r}} d_{c_i}^{e_i}$. Therefore, we can apply a "sieving method" based on each of the components individually. This means that we can cross out all of those possibilities which we know a priori that at least one of their components is zero and consequently the corresponding term is zero. This yields to reduce the number of possible terms which we have to construct, and as a result the procedure will speed up. We apply now in our method the following criteria.

Criterion 6.1.17
If $a_s = 0$, then all the partitions of m of length s do not contribute to the construction of the equation of $d_{r+1} = d_{m-\mathrm{val}+1}$.

Comment 6.1.18
It is important to point out that we apply this criterion before the computation of the partitions. We create a list lengths $:= [s : \ a_s \neq 0]$ *of the possible lengths of the partitions of m and then we compute only the partitions of m of length s, with $s \in$* lengths.

The second criterion is concerned with the product of d_{c_i}'s. Before presenting the criterion, we would like to recall the following definition.

Definition 6.1.19
A summand in a partition is also called a part.

Now we give the criterion.

Criterion 6.1.20
If $d_{c_i} = 0$ for some i, then we cross out all the partitions which have a part equal to c_i.

Note that we compute the d_{c_i}'s successively. After the computation of each d_{c_i} we check the current set of the remaining partitions and we did not compute those which have a part equal to c_i, when $d_{c_i} = 0$. In particular, we create a list, called allow, which contains the numbers c_i's such that $d_{c_i} \neq 0$, where $1 \leq c_i \leq m$, that is

$$\text{allow} := [c_i : \ 1 \leq c_i \leq m \text{ and } d_{c_i} \neq 0]$$

whose elements can be used as the possible candidates for the parts of the partitions of m. After the computation of every d_j we update the list allow. We take advantage of this list before the computation of the partitions of m. This means that, this list helps us to compute the partitions of m only in the essential cases.

The third main criterion is related to the coefficient w of the corresponding term of a partition.

Criterion 6.1.21
Let σ be a partition of m and w_σ be the coefficient of the corresponding term of σ. If $w_\sigma \equiv 0 \mod p$, then the partition σ does not contribute to the construction of the equation.

This criterion is the most complicated step, since we would like to find a way so as to know in advance when a partition has a non-zero w-value modulo p.

Remark 6.1.22
For the determination of w we compute the multinomial coefficient over the integers and after that we take $w \mod p$.

Now we are going to describe that way. Firstly, we analyze the p-valuation of the coefficient w.

Lemma 6.1.23
Let $m := \sum_{i=1}^{\bar{r}} c_i e_i$ and $s := \sum_{i=1}^{\bar{r}} e_i$. Then, $v_p(w) = \sum_{i\geq 1} \left(\left\lfloor \frac{s}{p^i} \right\rfloor - \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor \right)$.

Proof. By the definition of w, (see Remark 6.1.12), we have that $w = \frac{s!}{e_1!e_2!\cdots e_{\bar{r}}!}$. Thus, $v_p(w) = v_p(s!) - v_p(e_1!e_2!\cdots e_{\bar{r}}!)$. Also, by Legendre's Formula [24], we know that

$$v_p(s!) = \sum_{i\geq 1} \left\lfloor \frac{s}{p^i} \right\rfloor.$$

Then,

$$v_p(e_1!e_2!\cdots e_{\bar{r}}!) = \left\lfloor \frac{e_1}{p} \right\rfloor + \left\lfloor \frac{e_2}{p} \right\rfloor + \cdots + \left\lfloor \frac{e_{\bar{r}}}{p} \right\rfloor + \cdots + \left\lfloor \frac{e_1}{p^{\bar{r}}} \right\rfloor + \left\lfloor \frac{e_2}{p^{\bar{r}}} \right\rfloor + \cdots + \left\lfloor \frac{e_{\bar{r}}}{p^{\bar{r}}} \right\rfloor = \sum_{i\geq 1} \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor.$$

Thus,

$$v_p(w) = \sum_{i\geq 1} \left(\left\lfloor \frac{s}{p^i} \right\rfloor - \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor \right).$$

□

Remark 6.1.24

In fact, we have that $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor \leq \left\lfloor \frac{s}{p^i} \right\rfloor$ *for every* $i \geq 1$.

Lemma 6.1.25

$v_p(w) > 0$ *if and only if it exists at least one* i *such that* $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor < \left\lfloor \frac{s}{p^i} \right\rfloor$.

Thus, $v_p(w) = 0$ *if and only if* $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor = \left\lfloor \frac{s}{p^i} \right\rfloor$, *for every* $i \geq 1$.

Proof. According to Lemma 6.1.23, we have that

$$v_p(w) > 0 \Leftrightarrow \sum_{i\geq 1} \left(\left\lfloor \frac{s}{p^i} \right\rfloor - \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor \right) > 0.$$

Equivalently, it exists at least one i such that $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor < \left\lfloor \frac{s}{p^i} \right\rfloor$, due to Remark 6.1.24. □

Corollary 6.1.26

We have that $w \equiv 0 \mod p$ *if and only if it exists at least one* i *such that* $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor < \left\lfloor \frac{s}{p^i} \right\rfloor$.

Corollary 6.1.27

We have that $w \not\equiv 0 \mod p$ *if and only if* $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor = \left\lfloor \frac{s}{p^i} \right\rfloor$, *for every* $i \geq 1$.

According to Remark 6.1.12, we know that the w-value depends only on s and the e_i's, where s is one of the possible length of the partitions of m. Thus, the key idea is to compute and store the partition of s with non-zero w-value modulo p, for every possible length $s \in$ lengths. As a result, we will be able to use those information as many times as we need them through the computation of the image $\sigma(u)$. Thereafter, for those partitions of s we will compute the corresponding partitions of m. That is to say, for each partition of s with non-zero w-value modulo p, we will compute the possible parts c_i's, which in turn construct the corresponding partitions of m. In this way, we reduce the problem of computation of the partitions of m with non-zero w-value modulo p to that of the partitions of s. Therefore, we would like to compute the partitions of s with non-zero w-value modulo p. It is worth pointing out that for every partition of s we use the list allow of the possible values of c_i's in order to check if it is possible to construct the partitions of m. If so, then we compute the partitions of m which are derived by the partition of s.

For the determination of the partitions of s, we need a leading partition, which is defined below.

Definition 6.1.28

Let $s := n_a p^a + n_{a-1} p^{a-1} + \cdots, n_1 p + n_0$, *with* $0 \leq n_i \leq p-1$, *be the* p*-adic expansion of* s. *Then, the partition*

$$\tau_0 := [\underbrace{p^a, \cdots, p^a}_{n_a}, \underbrace{p^{a-1}, \cdots, p^{a-1}}_{n_{a-1}}, \dots, \underbrace{p, \cdots, p}_{n_1}, \underbrace{1, \cdots, 1}_{n_0}]$$

is called the leading partition of s.

We can prove that the leading partition has a non-zero w-value.

Lemma 6.1.29

The leading partition τ_0 *of* s *has a non-zero* w*-value.*

Proof. By Corollary 6.1.27, it suffices to show that $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor = \left\lfloor \frac{s}{p^i} \right\rfloor$, for every $1 \leq i \leq a$.
For $i = a$, we have that: $\left\lfloor \frac{s}{p^a} \right\rfloor = n_a = \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^a} \right\rfloor$.
For $i = a-1$, we have that: $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^{a-1}} \right\rfloor = n_{a-1} + n_a p$. Also,

$$\begin{aligned} \left\lfloor \frac{s}{p^{a-1}} \right\rfloor &= \left\lfloor \frac{n_a p^a + n_{a-1}p^{a-1} + \cdots, n_1 p + n_0}{p^{a-1}} \right\rfloor = \left\lfloor \underbrace{n_a p + n_{a-1}}_{\in \mathbb{Z}} + \frac{n_{a-2}p^{a-2} + \cdots, n_1 p + n_0}{p^{a-1}} \right\rfloor \\ &= n_a p + n_{a-1} + \left\lfloor \frac{n_{a-2}p^{a-2} + \cdots, n_1 p + n_0}{p^{a-1}} \right\rfloor . \end{aligned}$$

Since $0 \leq n_i \leq p-1$, then $0 \leq p^i n_i \leq p^i(p-1)$, for every $1 \leq i \leq a-2$. So then

$$0 \leq n_0 + n_1 p + \cdots + n_{a-2}p^{a-2} \leq (p-1)(1 + p + p^2 + \ldots + p^{a-2}) \Rightarrow$$
$$0 \leq n_0 + n_1 p + \cdots + n_{a-2}p^{a-2} \leq (p-1)\frac{1-p^{a-1}}{1-p} \Rightarrow 0 \leq n_0 + n_1 p + \cdots + n_{a-2}p^{a-2} \leq p^{a-1} - 1.$$

Hence, $\left\lfloor \frac{n_{a-2}p^{a-2} + \cdots, n_1 p + n_0}{p^{a-1}} \right\rfloor = 0$, and thus $\left\lfloor \frac{s}{p^{a-1}} \right\rfloor = n_a p + n_{a-1} = \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^{a-1}} \right\rfloor$. Similarly, we prove that

$$\left\lfloor \frac{s}{p^i} \right\rfloor = n_a p^{a-i} + n_{a-1}p^{a-1-i} + \cdots + n_{i+1}p + n_i = \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor, \quad \text{for every} \quad 1 \leq i \leq a-2.$$

Therefore, we have that $\sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor = \left\lfloor \frac{s}{p^i} \right\rfloor$, for every $1 \leq i \leq a$ and consequently $w_{\tau_0} \neq 0$. □

Definition 6.1.30
A partition of s is called a sub-sum partition of the leading partition τ_0 if its parts are sums of the parts of τ_0.

Remark 6.1.31
Let $p^\delta \mid\mid (s - (s \mod p))$. Then, for each part e_i of τ_0, we have that

$$e_i \equiv 0 \text{ or } s \mod p^\delta.$$

Moreover, each part $\tilde{e}_i$ of a sub-sum partition of the leading partition τ_0 satisfies that

$$\tilde{e}_i \equiv 0 \text{ or } s \mod p^\delta.$$

We give an example in order to explain the Remark 6.1.31.

Example 6.1.32
Let $s = 46 = 1 \cdot 27 + 2 \cdot 9 + 1$ and $p = 3$. Then, the leading partition is $\tau_0 = [27, 9, 9, 1]$.
So, $s \mod p \equiv 46 \mod 3 = 1$ and then

$$s - (s \mod p) = 45 = 5 \cdot 3^2 \Rightarrow 3^2 \mid\mid [s - (s \mod p)] \Rightarrow p^\delta = 3^2.$$

Thus, $s \mod p^\delta \equiv 46 \mod 3^2 = 1$ and then by Remark 6.1.31 we have that $e_i \equiv 0$ or $1 \mod 3^2$, for every $e_i \in \tau_0$. Indeed: $27 \equiv 0 \mod 3^2, 9 \equiv 0 \mod 3^2, 1 \equiv 1 \mod 3^2$.
In addition, this is true for every sub-sum partition of τ_0. In this case the sub-sums partitions are

$$[46], \quad [45, 1], \quad [37, 9], \quad [36, 10], \quad [28, 18], \quad [27, 19],$$
$$[36, 9, 1], \quad [28, 9, 9], \quad [27, 18, 1], \quad [27, 10, 9], \quad [27, 9, 9, 1].$$

Let now $\sigma = [27, 9, 3, 3, 3, 1]$. The partition σ is not a sub-sum partition of τ_0 and the claim of Remark 6.1.31 is not true.

Now we determine the desirable partitions of s.

Lemma 6.1.33
All partitions of s with corresponding non-zero w-value modulo p are sub-sums of the leading partition τ_0. Namely, the partitions whose parts are sums of the parts of the leading partition.

Proof. Firstly, we check that $w \not\equiv 0 \mod p$ for all sub-sums partitions of τ_0. We know that

$$\left\lfloor \frac{a}{p^i} \right\rfloor + \left\lfloor \frac{b}{p^i} \right\rfloor \leq \left\lfloor \frac{a+b}{p^i} \right\rfloor. \tag{6.21}$$

Let $\tilde{\tau}_s = [\tilde{e}_1, \dots, \tilde{e}_{\tilde{r}}]$, with $\tilde{r} < \bar{r}$, be a sub-sum partition of τ_0. Then,

$$\sum_{j=1}^{\tilde{r}} \left\lfloor \frac{\tilde{e}_j}{p^i} \right\rfloor \overset{(6.21)}{\geq} \sum_{j=1}^{\bar{r}} \left\lfloor \frac{e_j}{p^i} \right\rfloor = \left\lfloor \frac{s}{p^i} \right\rfloor$$

and by Remark 6.1.24 we have that $\sum_{j=1}^{\tilde{r}} \left\lfloor \frac{\tilde{e}_j}{p^i} \right\rfloor = \left\lfloor \frac{s}{p^i} \right\rfloor$, for every $1 \leq i \leq \tilde{r}$. Thus, $w_{\tau_s} \not\equiv 0 \mod p$. Therefore, w-values are non-zero, for every sub-sum partition of τ_0.
Now we would like to prove that those partitions are the only ones with non-zero w. This means that all the partitions of s which have not been constructed like above have $w \equiv 0 \mod p$.
Suppose that σ is a partition of s which is not a sub-sum of τ_0. Let $\sigma := [\hat{e}_1, \dots, \hat{e}_{\hat{r}}]$, where $\hat{r} \geq \bar{r}$ or $\hat{r} < \bar{r}$. It suffices to show that it exists at least one i such that $\left\lfloor \frac{s}{p^i} \right\rfloor > \sum_{j=1}^{\hat{r}} \left\lfloor \frac{\hat{e}_j}{p^i} \right\rfloor$.
We know that $s = \hat{e}_1 + \dots + \hat{e}_{\hat{r}}$ and it exists at least one $1 \leq j_0 \leq \hat{r}$ such that $\hat{e}_{j_0} \not\equiv 0$ or $s \mod p^\delta$, where $p^\delta \mid\mid (s - (s \mod p))$, by Remark 6.1.31. Also, $\hat{e}_j := \sum_{i=1}^{\bar{r}} \lfloor \xi_i^{\hat{e}_j} e_i \rfloor$, with $0 \leq \xi_i^{\hat{e}_j} \leq 1$, and $e_i = \sum_{j=1}^{\hat{r}} \lfloor \xi_i^{\hat{e}_j} e_i \rfloor$, for every $1 \leq i \leq \bar{r}$. Let $e_{i_0} = p^l$ such that $e_{i_0} = \sum_{j=1}^{\hat{r}} \lfloor \xi_{i_0}^{\hat{e}_j} e_{i_0} \rfloor$, where $0 \leq \xi_{i_0}^{\hat{e}_j} < 1$. Then, we have that

$$\sum_{j=1}^{\hat{r}} \left\lfloor \frac{\hat{e}_j}{p^l} \right\rfloor < \sum_{i=1}^{\bar{r}} \left\lfloor \frac{e_i}{p^l} \right\rfloor = \left\lfloor \frac{s}{p^l} \right\rfloor,$$

since $\hat{e}_{j_0} \not\equiv 0$ or $s \mod p^\delta$, for at least one $1 \leq j_0 \leq \hat{r}$. This means that it is not possible to reconstruct the term e_{i_0}, which was decomposed. Consequently, we lose one time the term p^l in $\sum_{j=1}^{\hat{r}} \left\lfloor \frac{\hat{e}_j}{p^l} \right\rfloor$. Otherwise, the resulting σ partition will be a sub-sum partition of τ_0, which is a contradiction by the definition of σ. Thus, $w_\sigma \equiv 0 \mod p$, by Corollary 6.1.26. Therefore, all the partitions of s which are not sub-sums of τ_0 have $w \equiv 0 \mod p$, which completes the proof. □

Wildly Ramified Case $p \mid \mathrm{val}$:

By equation (6.4), we obtain again the equation of d_1 of the form $a_{\mathrm{val}}\, d_1^{\mathrm{val}} = b_{\mathrm{val}}$. It is clear that for every $r \geq 1$ the denominator of the formula (6.7), i.e.

$$d_{r+1} = \frac{b_{r+\mathrm{val}} - \tilde{e}_{r+\mathrm{val}}}{\mathrm{val}\ a_{\mathrm{val}}\ d_1^{\mathrm{val}-1}},$$

is divisible by 0 in the case $p \mid \mathrm{val}$. Thus, the equation (6.6) yields that $\tilde{e}_{r+\mathrm{val}} = b_{r+\mathrm{val}}$, which means that it does not give any information for the determination of the d_{r+1}. However, in this case we know that $p \nmid \mathrm{val} + 1$, so then we repeat the above computations and determine the value of d_{r+1} by the equation

$$\mathfrak{F} := \mathrm{Expression}(a_1, \dots, a_j, d_1, \dots, d_{j-\mathrm{val}+1}) = b_j, \quad \text{for some } j > r + \mathrm{val}.$$

Proposition 6.1.34
Let $p \mid \mathrm{val}$. Then for every $r \geq 1$ we determine the value of d_{r+1}:
If $\mathrm{val} = 2$ and $r = 1$, then d_{r+1} is determined by the equation

$$\tilde{e}_{r+\mathrm{val}+1} + a_{r+\mathrm{val}+1}\, d_1^{r+\mathrm{val}+1} + (\mathrm{val}+1)\, a_{\mathrm{val}+1}\, d_1^{\mathrm{val}}\, d_{r+1} + a_{\mathrm{val}} d_{r+1}^{\mathrm{val}} = b_{r+\mathrm{val}+1}.$$

Otherwise, d_{r+1} is determined by the equation

$$\tilde{e}_{r+\mathrm{val}+1} + a_{r+\mathrm{val}+1}\, d_1^{r+\mathrm{val}+1} + (\mathrm{val}+1)\, a_{\mathrm{val}+1}\, d_1^{\mathrm{val}}\, d_{r+1} = b_{r+\mathrm{val}+1},$$

where $\tilde{e}_{r+\mathrm{val}+1} \in \mathbb{F}_q[d_1, \dots, d_r, d_{r+1}]$.

Proof. By Lemma 6.1.7, the equation $\sigma(\alpha_1) = \alpha_2$ implies

$$\alpha_1(S_{r+2}(u)) \equiv \alpha_2(u) \mod u^{r+\mathrm{val}+2}.$$

According to the proof of Lemma 6.1.7, for each $r \geq 1$ we have the following equation

$$\alpha_1(S_{r+2}(u)) \equiv \left(\sum_{i=\mathrm{val}}^{r+\mathrm{val}+1} a_i S_{r+1}^i(u) \right) + \mathrm{val}\, a_{\mathrm{val}}\, d_1^{\mathrm{val}-1}\, d_{r+2} u^{r+\mathrm{val}+1} \mod u^{r+\mathrm{val}+2}.$$

By analyzing the above equation, as in the proof of Lemma 6.1.7, we get that

$$\alpha_1(S_{r+2}(u)) \equiv \left(\sum_{i=\mathrm{val}}^{r+\mathrm{val}-1} e_i u^i \right) + \tilde{e}_{r+\mathrm{val}}\, u^{r+\mathrm{val}} + \tilde{e}_{r+\mathrm{val}+1}\, u^{r+\mathrm{val}+1} + a_{r+\mathrm{val}+1}\, d_1^{r+\mathrm{val}+1}\, u^{r+\mathrm{val}+1}$$
$$+(\mathrm{val}+1)\, a_{\mathrm{val}+1}\, d_1^{\mathrm{val}}\, d_{r+1} u^{r+\mathrm{val}+1} + a_{\mathrm{val}} d_{r+1}^{\mathrm{val}} u^{\mathrm{val}(r+1)} \mod u^{r+\mathrm{val}+2},$$

where $\tilde{e}_{r+\mathrm{val}} \in \mathbb{F}_q[d_1, \dots, d_r]$ and $\tilde{e}_{r+\mathrm{val}+1} \in \mathbb{F}_q[d_1, \dots, d_r, d_{r+1}]$.
Therefore, for every $r \geq 1$ we have that

$$\left(\sum_{i=\mathrm{val}}^{r+\mathrm{val}-1} e_i u^i \right) + \tilde{e}_{r+\mathrm{val}}\, u^{r+\mathrm{val}} + \tilde{e}_{r+\mathrm{val}+1}\, u^{r+\mathrm{val}+1} + a_{r+\mathrm{val}+1}\, d_1^{r+\mathrm{val}+1}\, u^{r+\mathrm{val}+1}$$
$$+(\mathrm{val}+1)\, a_{\mathrm{val}+1}\, d_1^{\mathrm{val}}\, d_{r+1} u^{r+\mathrm{val}+1} + a_{\mathrm{val}} d_{r+1}^{\mathrm{val}} u^{\mathrm{val}(r+1)} \equiv \sum_{i=\mathrm{val}}^{r+\mathrm{val}+1} b_i u^i \mod u^{r+\mathrm{val}+2}.$$

This yields the following equation of d_{r+1}

$$\tilde{e}_{r+\mathrm{val}+1} + a_{r+\mathrm{val}+1}\, d_1^{r+\mathrm{val}+1} + (\mathrm{val}+1)\, a_{\mathrm{val}+1}\, d_1^{\mathrm{val}}\, d_{r+1} = b_{r+\mathrm{val}+1} \tag{6.22}$$

if $\mathrm{val} \geq 3$ and $r \geq 1$ or $\mathrm{val} = 2$ and $r \geq 2$. For the case $\mathrm{val} = 2$ and $r = 1$, the equation of d_{r+1} is given by the following equation

$$\tilde{e}_{r+\mathrm{val}+1} + a_{r+\mathrm{val}+1}\, d_1^{r+\mathrm{val}+1} + (\mathrm{val}+1)\, a_{\mathrm{val}+1}\, d_1^{\mathrm{val}}\, d_{r+1} + a_{\mathrm{val}} d_{r+1}^{\mathrm{val}} = b_{r+\mathrm{val}+1}. \tag{6.23}$$

□

By equations (6.22) and (6.23), we can state a remark for the degree of d_{r+1} in this equation.

Comment 6.1.35
The degree of the equation (6.22) *or* (6.23) *with respect to d_{r+1} depends on the partitions of the number $r + \mathrm{val} + 1$ which contain the number $r + 1$.*

Remark 6.1.36
If the equation (6.22) or (6.23) does not give us information on determining the d_{r+1}, then we repeat the above computations and we will determine the value of d_{r+1} by the equation

$$\mathfrak{F} := \mathrm{Expression}(a_1, \dots, a_j, d_1, \dots, d_{j-\mathrm{val}+1}) = b_j, \quad \textit{for some} \quad j > r + \mathrm{val} + 1.$$

Therefore, in the case $p \mid \mathrm{val}$ we can determine the value of d_{r+1} by using the partitions of the number m, with $m \geq r + \mathrm{val} + 1$, for every $r \geq 1$.

For simplicity, we suppose that $\mathrm{val} = 2$ which means that $a_1 = 0 = b_1$ and $p = 2 \mid \mathrm{val}$. We will describe the above procedure of the determination of d_i, for each $i \geq 1$. We compute the equation of d_r by the equation

$$\alpha_1(S_r(u)) \equiv \alpha_2(u) \mod u^{r+val}$$

and we successively obtain the following system of equations

$$\begin{aligned}
a_2 d_1^2 &= b_2 && (6.24)\\
a_3 d_1^3 &= b_3 && (6.25)\\
a_2 d_2^2 + 3a_3 d_1^2 d_2 + a_4 d_1^4 &= b_4 && (6.26)\\
3a_3 d_1 d_2^2 + 3a_3 d_1^2 d_3 + a_5 d_1^5 &= b_5 && (6.27)\\
a_2 d_3^2 + 3a_3 d_1^2 d_4 + a_3 d_2^3 + 5a_5 d_1^4 d_2 + a_6 d_1^6 &= b_6 && (6.28)\\
&\vdots
\end{aligned}$$

By equation (6.24), we can determine the value of d_1. The equation (6.26) leads to a quadratic equation of d_2, since $a_2 \not\equiv 0 \mod 2$. In order to determine the value of d_3, we should distinguish two cases. If $a_3 \not\equiv 0 \mod 2$, then we can determine the value of d_3 by the equation (6.27) and the value of d_4 by the equation (6.28). Otherwise, we will determine the value of d_3 by the equation (6.28) and especially the equation (6.28) leads to a quadratic equation of d_3. For the determination of d_4 we will use the next equation. Similarly, we will determine the value of d_i for every $i \geq 5$.

An important point to note here is that in the $p \mid \mathrm{val}$ case we will not encounter the situation in which an equation will consist of more than one undefined d_i's.

Let us assume that using the partitions of a number $m \in \mathbb{N}$ we construct the corresponding equation with respect to our approach (see Proposition 6.1.15). We will denote it by $\mathfrak{F} = b_m$. Note that if that equation has no solution or infinitely many solutions, then we proceed to the next m, and repeat until we find a non-trivial one. Without loss of generality we assume that this equation is a non-trivial one. We know that $\mathfrak{F}$ is composed of terms, and in turn each term consist of three components, which are w, a_s and the product $\prod_{i=1}^{\bar{r}} d_{c_i}^{e_i}$, according to Remark 6.1.12. Additionally, it holds that it would be appeared d_j with $1 \leq j \leq m$ and a_s with $1 \leq s \leq m$.

We suppose that performing successively our procedure we have computed the values of $d_1, \ldots, d_i$ with $1 \leq i < j \leq m$. Therefore, we aim to determine d_{i+1}.

Lemma 6.1.37
Under the above assumptions, the equation $\mathfrak{F} = b_m$ is an equation in terms of $d_1, \ldots, d_{i+1}$, which yields a non-trivial equation for d_{i+1}.

Proof. We begin by studying the structure of the equation $\mathfrak{F} = b_m$. Considering the description of our method, we know that the terms consisting of d_j with $i < j \leq m$ also include a value w and a coefficient a_s where $1 \leq s < j$ (especially, it holds that $1 \leq s < j - 1$).
Regarding the terms including d_j with $i + 1 < j \leq m$ we have that either $w \equiv 0 \mod p$ or $a_s = 0$ with $1 \leq s < j$, by definition. It is worth pointing out that the coefficients a_s are contained in the terms consisting of d_j with $i + 1 < j \leq m$ will be for "small" s, that is to say $1 \leq s < j - 1$. Moreover, the aforementioned a_s with $1 \leq s < j - 1$ will have already appeared in the previous equations. This means that they will be known to us, and either $a_s = 0$ or $w \equiv 0 \mod p$ in the corresponding term. Thus, in this case the terms consisting of d_j with $i+1 < j \leq m$ are eliminated.
It remains to analyze the terms containing d_{i+1}. As described above, the terms consisting of d_{i+1} also include a value w and a coefficient a_s where $1 \leq s < i + 1$. In this case, the equation $\mathfrak{F} = b_m$ leads to an equation in terms of d_{i+1}, which in turn has solutions, which completes the proof. □

In $p \mid \mathrm{val}$ case it is natural to ask whether an equation in terms of d_i can be constructed for every i. This is why the following remark is given.

Remark 6.1.38
For the determination of d_i, we repeat the procedure up to the point we reach the equation which involves the term of the form $a_{\mathrm{val}} d_i^{\mathrm{val}}$. In particular, the aforementioned equation will be constructed by the partitions of the number $\tilde{m} = i \cdot \mathrm{val}$. Thus, the equation is $\mathfrak{F} = b_{\tilde{m}}$.

Comment 6.1.39
We emphasize that the Remark 6.1.38 describes the worst case scenario. Generally speaking, we look for an equation which contains a non-zero term of the form

$$w a_{\mathrm{val}} d_i^{\kappa} \prod_l d_l, \quad \textit{for some} \quad \kappa \quad \textit{and } l \quad \textit{such that } l < i.$$

In both cases when we perform the process determining $\sigma(u)$ such that $\sigma(\alpha_1) = \alpha_2$ we should be able to construct as many different such automorphisms σ as $[N : K(\alpha_1)]$.

Assume that $f \in K[X]$ is an Eisenstein polynomial. In the case $p \nmid \mathrm{val}$, by Proposition 6.1.8, we have that the equation of d_1 is of degree val. Also, once we have chosen the value of d_1, the value of d_{r+1} is uniquely determined for every $r \geq 1$. Thus, we have $\mathrm{val} = v_u(\alpha)$ possible values of d_1, and consequently we can construct $\mathrm{val} = [N : K(\alpha_1)]$ automorphisms σ such that $\sigma(\alpha_1) = \alpha_2$.
The case $p \mid \mathrm{val}$ is more complicated. Let $\mathrm{val} := p^{v_p(\mathrm{val})} q$, where $\gcd(p, q) = 1$. Then we have q values for d_1, which means that it suffices to determine in which d_i's we get the rest.

In case $p \nmid \text{val}$ we know in advance the degree of the equation of d_r for every $r \geq 1$. However, this is not true when $p \mid \text{val}$. It is of interest to estimate the degree of the equation of d_r for every $r \geq 1$ in the latter case as well. To do so, we give the following heuristic result for Eisenstein polynomials. For simplicity of notation, we denote by g_{d_r} the polynomial which corresponds to the equation of d_r for every $r \geq 1$.

Remark 6.1.40
For Eisenstein polynomials we make a remark about the degree of the equation of d_r for every $r \geq 1$, which is based on our experiments. Let us denote by $B := [b_1, b_2, b_3, \ldots, b_\kappa]$ the breaks in which $d_{b_1}, d_{b_2}, d_{b_3}, \ldots, d_{b_\kappa}$ have distinct values, and in this way we ensure the construction of val *automorphisms σ such that $\sigma(\alpha_1) = \alpha_2$. Additionally, let $\rho(g_{d_{b_i}})$ stands for the number of distinct roots of $g_{d_{b_i}}$. Then we noticed that the following pattern holds for the degrees:*

$$\begin{aligned}
\deg(g_{d_r}) &= \text{val}, && \textit{for } 1 \leq r \leq b_1,\\
\deg(g_{d_r}) &= \frac{\text{val}}{\rho(g_{d_{b_1}})}, && \textit{for } b_1 < r \leq b_2,\\
\deg(g_{d_r}) &= \frac{\deg(g_{d_{b_2}})}{\rho(g_{d_{b_2}})}, && \textit{for } b_2 < r \leq b_3,\\
&\vdots\\
\deg(g_{d_r}) &= \frac{\deg(g_{d_{b_{\kappa-1}}})}{\rho(g_{d_{b_{\kappa-1}}})}, && \textit{for } b_{\kappa-1} < r \leq b_\kappa,\\
\deg(g_{d_r}) &= \frac{\deg(g_{d_{b_\kappa}})}{\rho(g_{d_{b_\kappa}})} = 1, && \textit{for } r \geq b_\kappa.
\end{aligned}$$

Moreover, it is worth noting the following statement.

Comment 6.1.41
Let us assume that we construct g_{d_r} by using the permutations of the number $m \in \mathbb{N}$. Then the polynomial $g_{d_{r+1}}$ will be constructed by using the permutations of the number $m + \deg(g_{d_r})$.

It should be noted that the list $B := [b_1, b_2, b_3, \ldots, b_\kappa]$ of breaks is important for the construction of the automorphisms for any polynomial, which is explained in the following remark.

Remark 6.1.42
Let us denote by $B := [b_1, b_2, b_3, \ldots, b_\kappa]$ the breaks in which $d_{b_1}, d_{b_2}, d_{b_3}, \ldots, d_{b_\kappa}$ have distinct values. Additionally, let δ_i stands for the number of distinct roots of $g_{d_{b_i}}$ with $1 \leq i \leq \kappa$, and define $\Delta := [\delta_1, \ldots, \delta_\kappa]$. We can construct $\delta_1 \cdots \delta_\kappa$ different automorphisms, and in this way we ensure the construction of all automorphisms σ such that $\sigma(\alpha_1) = \alpha_2$. The tuple Δ is the upper bound of the possible tuples Γ which can be used for the determination of automorphisms applying our approach. In particular,

$$\Gamma = [\gamma_1, \ldots, \gamma_\kappa] \textit{ such that } 1 \leq \gamma_i \leq \delta_i, \textit{ with } 1 \leq i \leq \kappa.$$

Remark 6.1.43
We note that Lemma 6.1.2 states that after applying our method to determine $\sigma(u)$ it is needed to check that $\sigma|_K = \text{id}$ in order to ensure that σ gives rise to an element of the desired Galois group.

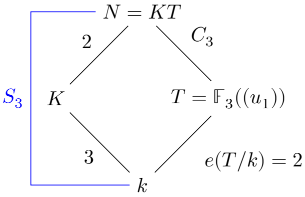
Let us now outline our approach to compute the desired automorphisms. We begin by determining the automorphism σ as described above. To be more precise, using the aforementioned method we define an automorphism σ of N which maps α_i to α_j as follows

$$\sigma: \ N \to N, \ \text{ such that } \sigma(u) = \sum_{\kappa \geq 1} d_\kappa u^\kappa.$$

We are left with the task of checking that σ is a K-automorphism of N, i.e. $\sigma|_K = \text{id}$, where $K = \mathbb{F}_q((t))$. It is obvious that $\sigma|_{\mathbb{F}_q} = \text{id}$, since $\mathbb{F}_q \subseteq \mathbb{F}_{\tilde{q}}$ and $\sigma|_{\mathbb{F}_{\tilde{q}}} = \text{id}$, as $\mathbb{F}_{\tilde{q}}$ is the prime subfield of N. Therefore, we only need to check that $\sigma(t) = t$. If so, we store σ for the computation of the Galois group and continue to the next step. Otherwise, we discard it and repeat the computation of an automorphism in the same step. In particular, we repeat the automorphism by using a different tuple Γ for the choice of the values of d_{c_i}'s with distinct values.

We give an example to demonstrate the importance of the condition that $\sigma|_K = \text{id}$.

Example 6.1.44
Let $f(X) = X^3 + tX + t \in k[X]$, where $k = \mathbb{F}_3((t))$. It is clear that f is irreducible, and it is easily seen that $[N : k] = 6$ and $\mathrm{Gal}(f/k) \cong S_3$.



It holds that $T = \mathbb{F}_3((u_1))$ such that $\Phi_1(t) = 2u_1^2$. On the other hand, if we consider $f \in T[X]$, then it is easy to check that $[N : T] = 3$ and $\mathrm{Gal}(f/T) \cong C_3$.

Case 1: Consider $f \in k[X]$.
Applying our Splitting Field algorithm we find that $\mathrm{Spl}(f|k) = \mathbb{F}_3((u_2))$ with $\mathrm{prec} = 112$ and $[\mathrm{Spl}(f) : k] = 6$, its roots $\alpha_1, \alpha_2, \alpha_3 \in \mathrm{Spl}(f|k)$ and the embedding elements

$$\begin{aligned}\Phi_1(t) &= 2u_1^2 \mod u_1^{112}\\ \Phi_2(u_1) &= u_2^3 + u_2^5 + 2u_2^7 + u_2^9 + u_2^{17} + 2u_2^{19} + u_2^{21} + u_2^{23} + 2u_2^{25} + u_2^{27} + u_2^{53}\\ &\quad +2u_2^{55} + u_2^{57} + u_2^{59} + 2u_2^{61} + u_2^{63} + u_2^{71} + 2u_2^{73} + u_2^{75} + u_2^{77} + 2u_2^{79} + u_2^{81} \mod u_2^{112}.\end{aligned}$$

Thus, $\Phi_2(\Phi_1(t)) = 2u_2^6 + u_2^8 + u_2^{10} + 2u_2^{16} + 2u_2^{18} + u_2^{20} + 2u_2^{24} + u_2^{26} + u_2^{28} + 2u_2^{52} + 2u_2^{54} + u_2^{56} + 2u_2^{60} + u_2^{62} + u_2^{64} + 2u_2^{70} + 2u_2^{72} + u_2^{74} + 2u_2^{78} + u_2^{80} + u_2^{82} \mod u_2^{112}$.

Let M be the maximum u_2-valuation of the differences of the roots, i.e

$$M := \max\{v_{u_2}(\alpha_1 - \alpha_2), v_{u_2}(\alpha_1 - \alpha_3), v_{u_2}(\alpha_2 - \alpha_3)\} = 3.$$

So employing our method for the computation of the automorphisms which map α_1 to α_3 we get that the following two automorphisms $\sigma_1, \sigma_2 \in \mathrm{Aut}(Spl(f/k))$ such that

$$\sigma_1(u_2) = 2u_2 + 2u_2^2 \mod u_2^4$$
$$\sigma_2(u_2) = u_2 + u_2^2 \mod u_2^4.$$

Then, those automorphisms induce the permutations $(1\,3)$ and $(1\,3\,2)$ which are both elements of $\mathrm{Gal}(f/k)$. Additionally, we have that $\sigma_1(t) \equiv t \mod u_2^{M+1}$ and $\sigma_2(t) \equiv t \mod u_2^{M+1}$.

Case 2: Consider $f \in T[X]$.
This means that $f = X^3 + tX + t = X^3 + 2u_1^2X + 2u_1^2 \in T[X]$, as $\Phi_1(t) = 2u_1^2$. Applying our Splitting Field algorithm we find that $\mathrm{Spl}(f|T) = \mathbb{F}_3((u_2))$ with $\mathrm{prec} = 121$ and $[\mathrm{Spl}(f) : T] = 3$, its roots $\alpha_1, \alpha_2, \alpha_3 \in \mathrm{Spl}(f|T)$ and the embedding element

$$\begin{aligned}\Phi_2(u_1) = u_2^3 + u_2^5 + 2u_2^7 + u_2^9 + u_2^{17} + 2u_2^{19} + u_2^{21} + u_2^{23} + 2u_2^{25} + u_2^{27} + u_2^{53} + 2u_2^{55} + u_2^{57} + u_2^{59}\\ + 2u_2^{61} + u_2^{63} + u_2^{71} + 2u_2^{73} + u_2^{75} + u_2^{77} + 2u_2^{79} + u_2^{81} \mod u_2^{121}.\end{aligned}$$

Let M be the maximum u_2-valuation of the differences of the roots, i.e

$$M := \max\{v_{u_2}(\alpha_1 - \alpha_2), v_{u_2}(\alpha_1 - \alpha_3), v_{u_2}(\alpha_2 - \alpha_3)\} = 3.$$

So, employing our method for the computation of the automorphisms which map α_1 to α_3 we get that the following two automorphisms $\sigma_1, \sigma_2 \in \mathrm{Aut}(Spl(f/k))$ such that

$$\sigma_1(u_2) = 2u_2 + 2u_2^2 \mod u_2^4$$
$$\sigma_2(u_2) = u_2 + u_2^2 \mod u_2^4.$$

Similarly, those automorphisms give rise to the permutations $(1\,3)$ and $(1\,3\,2)$. The permutation $(1\,3\,2)$ is an element of $\mathrm{Gal}(f/T) \cong C_3$, and also σ_2 meets the condition that $\sigma_2(u_1) \equiv u_1 \mod u_2^{M+1}$. However, the permutation $(1\,3)$ is not an element of $\mathrm{Gal}(f/T)$. In particular, σ_1 fails to satisfy the assumptions of Lemma 6.1.2, since $\sigma_1(u_1) \not\equiv u_1 \mod u_2^{M+1}$, and thus it holds that $\sigma_1 \in \mathrm{Aut}(Spl(f/k))$ but $\sigma_1 \notin \mathrm{Aut}(Spl(f/T))$. In other words, σ_1 is not a k-automorphism.

For the sake of completeness the results and theory above are summarized in algorithmic forms, which are given below.

According to Proposition 6.1.15, we are able to construct the equation of d_i for every i and therefore we can determine the value of d_i for each i. To do so, we need to compute the partitions of $m := i + \mathrm{val} - 1$. For this reason, we would like to compute the partitions of s for every s, where s is a length of possible partitions of m, and consequently we need to compute the leading partition for each s. Firstly, we give the algorithm for the computation of the leading partition of s.

```
Algorithm 23: Leading partition [StartingPartition]
Input: A number $s$ and a prime number $p$.
Output: The leading partition $\tau_0$.
Compute the p-adic expansion of $s$, that is $s := n_a p^a + n_{a-1}p^{a-1} + \cdots, n_1 p + n_0$, with $0 \le n_i \le p-1$;
Construct the leading partition of $s$, $\tau_0 \leftarrow [\underbrace{p^a, \cdots, p^a}_{n_a}, \underbrace{p^{a-1}, \cdots, p^{a-1}}_{n_{a-1}}, \ldots, \underbrace{p, \cdots, p}_{n_1}, \underbrace{1, \cdots, 1}_{n_0}]$;
return $\tau_0$;
```

For the computation of all possible sub-sums partitions of the leading partition τ_0, we employ the algorithm called "SubPartitions". Given the leading partition

$$\tau_0 := [\underbrace{p^a, \cdots, p^a}_{n_a}, \underbrace{p^{a-1}, \cdots, p^{a-1}}_{n_{a-1}}, \ldots, \underbrace{p, \cdots, p}_{n_1}, \underbrace{1, \cdots, 1}_{n_0}],$$

that algorithm returns a list of all possible sub-sums partitions of it, (see Definition 6.1.30).

Now we present the algorithm of the computation of the partitions of s.

```
Algorithm 24: Compute partitions of s [Compute_s_Partitions]
Input: An integer $s$, a prime $p$, the list allow, and the coefficient $a_s$.
Output: A data-set of the elements $\langle \mathfrak{e}, w_\mathfrak{e} a_s, \mu_\mathfrak{e} \rangle$, where $\mathfrak{e}$ is a sub-sum partition of $\tau_0$, $w_\mathfrak{e}$ is its $w$-value and $\mu_\mathfrak{e}$
        its minimal value.
Compute the leading partition $\tau_0$ of $s$, that is $\tau_0 \leftarrow \mathrm{StartingPartition}(s, p)$;
Compute the list of the possible partitions of $s$, namely ListOfPart ← SubPartitions($\tau_0$);
$L_w \leftarrow []$;
foreach $\mathfrak{e} := [e_1, \ldots, e_r] \in \mathrm{ListOfPart}$ do
    $w_\mathfrak{e} \leftarrow \binom{s!}{e_1!,\ e_2!,\ \ldots,\ e_r!}$ mod $p$ and $\mu_\mathfrak{e} \leftarrow \sum_{i=1}^{|\mathfrak{e}|} e_i \mathrm{allow}[i]$;
    if $w \neq 0$ then $L_w \leftarrow L_w \cup \langle \mathfrak{e}, w_\mathfrak{e} a_s, \mu_\mathfrak{e} \rangle$;
return $L_w$;
```

For every partition of s we want to construct the possible partitions of m. Particularly, we want to compute the parts c_i's of the possible partitions of m, which are generated by the partitions of s.

By using Algorithm 24, we determine a list of the partitions of s, and denote it by $\mathrm{part}(s)$. Let $\mathfrak{e} := [e_1, \ldots, e_r]$ be a partition of s. So it holds that $s := \sum_{i=1}^{r} e_i$. Then we would like to compute all possible $C := [c_1, \ldots, c_r]$ such that $m = \sum_{i=1}^{r} c_i e_i$. In general, given $\mathfrak{e} \in \mathrm{part}(s)$, we will repeat that for every $m \le Z$, where $Z := \nu_{\max} + \mathrm{val} - 1$. The integer Z is the upper bound of applying our method for the determination of d_i's with $1 \le i \le \nu_{\max}$, where $\nu_{\max}$ is defined to be the maximum valuation of the differences of the roots. We explain this upper bound later in this section.

Since we highly use all those data throughout the determination of $\sigma(u)$, we store and reuse them. Especially, given a partition $\mathfrak{e} := [e_1, \ldots, e_r]$ of s, we compute the partitions $C := [c_1, \ldots, c_r]$ of m for every $m \le Z$ and store them.

To do so, we employ the algorithm called "WeightedRestrictedPartitions(n, W, A)", which has been implemented in C programming language. Given an integer n and the lists W, A of integers, it returns the partitions of κ restricted to elements of A, weighted with elements in W for every $1 \le \kappa \le n$. In our case $n := Z$, $W := \mathfrak{e}$ and $A := \mathrm{allow}$.

We emphasize that the idea of computing and storing the desired partitions C of m for every $1 \le m \le Z$ of a given partition $\mathfrak{e}$ of s accelerates the determination of $\sigma(u)$, since we compute more data with almost the same computational cost as performing the computations for the exact

m. Furthermore, it is worth pointing out that the implementation of the aforementioned idea in C programming language, particularly the "WeightedRestrictedPartitions(n, W, A)" algorithm, is a vital point in our procedure, since it reduces the computational cost of that kind of arithmetic. The algorithms "WeightedRestrictedPartitions(n, W, A)" and "SubPartitions" have been implemented by Markus Kirschmer, to whom we are grateful.

Once we have constructed the equation of each d_i, we can determine the image $\sigma(u) = \sum_{i\geq 1} d_i u^i$ of u such that $\sigma(\alpha_1) = \alpha_2$. The ideas described above about the image construction are summarized in an algorithmic form, which is given below.

Algorithm 25: Image construction [ConstructImage]

```
Input: The roots $\rho_1 = \sum_{i\geq \text{val}} a_i u^i$ and $\rho_2 = \sum_{i\geq \text{val}} b_i u^i$ of the polynomial, the splitting field $\text{Spl}(f) = \mathbb{F}_{\tilde{q}}((u))$,
       with $\tilde{q} = q^l$, of the polynomial, the needed size $\nu_{\max}$ and the tuple $\Gamma$.
Output: The image of $u$ such that $\sigma(\rho_1) = \rho_2$ and $B, \Delta, \Gamma$, (See Remark 6.1.42).
1  Define $\text{val} = v_u(\rho_1) = v_u(\rho_2)$, $\text{prec} \leftarrow \min(\text{prec}(\rho_1), \text{prec}(\rho_2))$, the upper bound for the size of the
     polynomial ring $\text{ub} \leftarrow \nu_{\max} + 8\text{val}$, $P \leftarrow \mathbb{F}_q[d_1, \dots, d_{\text{ub}}]$, $Z \leftarrow \nu_{\max} + \text{val} - 1$, $\text{allow} \leftarrow [1, \dots, \text{ub}]$;
2  Initialize $\Lambda_3(\text{im}) \leftarrow [\ ]$, $\kappa_b \leftarrow 1$, $i \leftarrow 1$, $B \leftarrow []$ and $\Delta \leftarrow []$, the stored list store_parts $\leftarrow [\ ]$ of the partitions
     of $s$, $\text{data}(s) \leftarrow [[] \ : \ 1 \leq j \leq Z]$ and the set zeropos $\leftarrow \{\ \}$ of the positions $i$ such that $d_{c_i} = 0$;
3  while $|\Lambda_3(\text{im})| \leq Z$ and $m \leq Z$ do
4      if $m = \text{val}$ and $\Lambda_0(\text{im}) = \emptyset$ then $\Lambda_0(\text{im}) \leftarrow [d_1, \dots d_m]$ and $D_1(\text{im}) \leftarrow [d_i]$ else $\Lambda_0(\text{im}) \leftarrow \Lambda_3(\text{im})$,
        $l \leftarrow |\Lambda_0(\text{im})|$, $\Lambda_0(\text{im}) \leftarrow \Lambda_0(\text{im}) \cup [d_l \dots d_i]$ and $D_1(\text{im}) \leftarrow \Lambda_3(\text{im}) \cup [d_i]$;
5      Initialize the list of partitions, part $\leftarrow [\ ]$ and $s \leftarrow m$;
6      if $a_s \neq 0$ then
7          part$(s) \leftarrow$ Compute_s_partitions$(s, p, \text{allow}, a_s)$ and store_parts $\leftarrow$ store_parts $\cup$ part$(s)$
8      part$(s) \leftarrow [\mathfrak{z} = \langle E, a_s \cdot w_E, \mu_E \rangle : \mathfrak{z} \in$ store_parts $\mid \mu_E = s]$ and
        store_parts $\leftarrow [\mathfrak{z} = \langle E, a_s \cdot w_E, \mu_E \rangle : \ \mathfrak{z} \in$ store_parts $\mid \mu_E > s]$;
9      foreach $\mathfrak{z} = \langle E, a_s \cdot w_E, \mu_E \rangle \in$ part$(s)$ do
10         list_c$_\text{i}$, list_m $\leftarrow$ WeightedRestrictedPartitions$(Z - 1, E, \text{allow})$;
11         for $j := 1$ to |list_c$_\text{i}$| do
12             $C \leftarrow$ list_c$_\text{i}[j]$, $\tilde{m} \leftarrow$ list_m$[j]$, and data_s$[\tilde{m}] \leftarrow$ data_s$[\tilde{m}] \cup [C, E, a_s w_E]$;
13     part $\leftarrow \text{data}(s)[m]$;
14     if |part| = 0 then
15         $g \leftarrow -b_m$, $m \leftarrow m + 1$ and go to Step 3. This case is possible when $p \mid \text{val}$;
16     else
17         $g \leftarrow (\sum_{\xi\in\text{part}} w a_s \prod_{i=1}^{r} (\Lambda[c_i])^{e_i}) - b_m = (\sum_{\xi\in\text{part}} w a_s \prod_{i=1}^{r} d_{c_i}^{e_i}) - b_m$, where $\xi = [C, E, w \cdot a_s]$ with
          $C := [c_1, \dots, c_r]$, $E := [e_1, \dots, e_r]$ such that $m = \sum_{i=1}^{r} c_i e_i$, and $s := \sum_{i=1}^{r} e_i$ and a list
          $\Lambda = D_1(\text{im}) = [d_1, \dots, d_i]$ of the coefficients of the partial image of $u$.
18     if $g = 0$ or $g \in \mathbb{F}_{\tilde{q}}$ then $m \leftarrow m + 1$ and go to Step 3. This case is possible when $p \mid \text{val}$;
19     Define $\hat{g} \in \mathbb{F}_{\tilde{q}}[X]$ obtained by $g \in \mathbb{F}_{\tilde{q}}[d_i]$;
20     if $\deg(\hat{g}) > 1$ then
21         Factorize the polynomial $\hat{g}$ and denote by $R(\hat{g})$ its roots;
22         if $|R(g)| = 1$ then pos $\leftarrow 1$;
23         else
24             if $|\Gamma| = 1$ then pos $\leftarrow \Gamma[1]$;
25             else
26                 pos $\leftarrow \Gamma[\kappa_b]$ and $\kappa_b \leftarrow \kappa_b + 1$;
27                 if $\kappa_b > |\Gamma|$ then $\kappa_b \leftarrow |R(\hat{g})|$;
28         $l$_im$_3 \leftarrow l$_im$_3 \cup [R(\hat{g})[\text{pos}]]$;
29         if $R(\hat{g})[\text{pos}] = 0$ then zeropos $\leftarrow$ zeropos $\cup \{|\Lambda_3(\text{im})|\}$ and allow $\leftarrow$ allow $\setminus [|\Lambda_3(\text{im})|]$;
30         if $|R(\hat{g})| > 1$ then $B \leftarrow B \cup [|\Lambda_3(\text{im})|]$ and $\Delta \leftarrow \Delta \cup [|R(\hat{g})|]$;
31     else
32         Define $\rho$ to be the root of $\hat{g}$ and $\Lambda_3(\text{im}) \leftarrow \Lambda_3(\text{im}) \cup [\rho]$;
33         if $\rho = 0$ then zeropos $\leftarrow$ zeropos $\cup \{|\Lambda_3(\text{im})|\}$ and allow $\leftarrow$ allow $\setminus [|\Lambda_3(\text{im})|]$;
34     $m \leftarrow m + 1$ and $i \leftarrow i + 1$;
35 Define the list $\Lambda(\text{im}) \leftarrow [0] \cup \Lambda_3(\text{im})$ and then take its corresponded element of $\text{Spl}(f)$ and denote it by im;
36 Change the precision of the element im, that is $\text{im} \leftarrow \text{im} \mod u^z$, where $z = \nu_{\max} + \text{val}$;
37 return im, $B, \Delta, \Gamma$;
```

The above algorithm takes as input the parameter Γ. As we have already mentioned, this variable helps us to choose the values of d_i's in each step. In case we should repeat the procedure of the image construction defining a new automorphism in the same step, the tuple Γ guarantees a new choice of each d_i if it exists. Otherwise, we choose the last known choice of the current d_i. Therefore, this variable guarantees to define different automorphisms in the same step providing that they exist.

To be more precise, under the same notation of Remark 6.1.42, the aforementioned tuple $\Gamma := [\gamma_1, \dots, \gamma_\kappa]$ guides us how to choose the values of d_{b_i} for $1 \leq i \leq \kappa$ in order to determine different automorphisms when needed.

For the determination of Γ the list of breaks B is necessary, and especially the list

$$\Delta := [\delta_1, \dots, \delta_\kappa].$$

Regarding the list Δ, we emphasize that we do not know it in advance. We do have access to it after one execution of our method. This is why in the first attempt of our method it is needed to make a guess for its length so as to be able to choose one random tuple Γ. Experiments drive us getting the following heuristic assumption concerning the length of Δ in the first attempt.

Heuristic Assumption 6.1.45
Under the same assumptions of Remark 6.1.42, let $B := [b_1, b_2, b_3, \dots, b_\kappa]$ be the breaks in which $d_{b_1}, d_{b_2}, d_{b_3}, \dots, d_{b_\kappa}$ have distinct values. Let also $v_u(\alpha) = p \cdot p_2 \cdots p_l$, and $n(S)$ denotes the number of slopes of the ramification polygon. Then,

$$\kappa = \min\{v_p(v_u(\alpha)), n(S)\}$$

and especially

$$\kappa = \begin{cases} \kappa + 1 & \textit{if } v_p(v_{u_\beta}(\alpha)) = 0 \textit{ or } l + 1 > 1 \\ \kappa & \textit{else} \end{cases}.$$

Next, for the first attempt, we initialize $\Gamma = [1, \dots, 1, 2]$ of length equals to κ, according to Heuristic Assumption 6.1.45. After the completion of the first try, we properly determine the lists $B = [b_1, b_2, b_3, \dots, b_\kappa]$ and $\Delta = [\delta_1, \delta_2, \delta_3, \dots, \delta_\kappa]$, and so we properly initialize the tuple $\Gamma := [\gamma_1, \dots, \gamma_\kappa]$ for the next try.

With the intention of computing the image $\sigma(u)$ of u such that $\sigma(\alpha_1) = \alpha_2$ it is sufficient to determine a “partial” image of u. This means that it is enough to compute the element

$$S_r(u) := \sum_{i=1}^{r} d_i u^i \in \mathbb{F}_q[u],$$

for an efficient number r. Therefore, we should figure out the precision of this element, that is the number r. We would like the number r being the minimal number that leads to a valid permutation of the roots. In order to compute the image $\sigma(u) = \sum_{i \geq \text{val}} d_i u^i$ we successively compute the “partial” images $S_{\tilde{r}}(u)$ for $\tilde{r} > 1$. Computing successively the partial images of u we find out that it exists a “critical” index r such that it gives a valid permutation of the roots. This means that the computation of $S_i(u)$ for every $i > r$ gives no new information. The new $S_i(u)$ for every $i \geq r$ yields the same results as those achieved by $S_r(u)$. We choose r to be the maximum valuation of the elements $\alpha_1 - \alpha_i$ with $2 \leq i \leq n$. According to our notation established above, we have that $r := \nu_{\max}$, and so

$$\tilde{\sigma}(u) = \sum_{i=1}^{\nu_{\max}} d_i u^i \equiv \sigma(u) \mod u^{\nu_{\max}+1}.$$

Corollary 6.1.46
Let $N := \operatorname{Spl}(f) = \mathbb{F}_{\tilde{q}}((u))$, $\alpha_1, \dots, \alpha_n$ be the roots of f and define $\sigma \in \operatorname{Aut}(N/K)$ under the same assumptions of Lemma 6.1.2. Let also define $\tilde{\sigma}$ as follows

$$\tilde{\sigma}(u) = \sum_{i=1}^{\nu_{\max}} d_i u^i \equiv \sigma(u) \mod u^{\nu_{\max}+1}.$$

Then, $\tilde{\sigma}$ induces a permutation of the roots of $f \in K[X]$.

Proof. Since $\sigma \in \operatorname{Aut}(N/K)$, it holds that $\tilde{\sigma} \in \operatorname{Aut}(N)$. It holds that $f(\tilde{\sigma}(\alpha_i)) \equiv 0 \mod u^{\nu_{\max}+1}$ for every $1 \leq i \leq n$ and $\tilde{\sigma}(\alpha_i) \not\equiv \tilde{\sigma}(\alpha_j) \mod u^{\nu_{\max}+1}$ for every $i \neq j$. Thus, the approximations of the roots can be distinguished from each other, and so $\tilde{\sigma}$ induces a permutation of the roots. □

We emphasize that for $p \mid v_u(\alpha)$ or $v_u(\alpha) > 1$ we can use smaller precision as indicated in the following remark.

Remark 6.1.47
Under the above assumptions, let $\alpha = \sum_{j \geq v_u(\alpha)} a_j u^j$ *be a root. Define*

$$M_\alpha := \min\left(\{j \ : \ v_u(\alpha) \leq j \leq \nu_{\max} \mid p \nmid j,\ a_j \neq 0\} \cup \{\nu_{\max}\}\right),\ \textit{and}$$

$$\tilde{B} := \max\{\nu_{\max} + 1 - M_\alpha, \lceil \frac{\nu_{\max}}{p} \rceil\} \leq \nu_{\max}.$$

Then it is sufficient to compute d_i *for every* $1 \leq i \leq \tilde{B}$.

Proof. As above $\sigma(u) = \sum_{i \geq 1} d_i u^i$ and so $\sigma(\alpha) = \sum_{j \geq v_u(\alpha)} a_j \sigma(u)^j = \sum_{j \geq v_u(\alpha)} a_j \left(\sum_{i \geq 1} d_i u^i\right)^j$. Let $d_{\tilde{B}+1}$ be the first unknown coefficient. Let $m = p^{v_p(m)} s \geq v_u(\alpha)$ with $p \nmid s \in \mathbb{N}$ such that $a_m \neq 0$. Then,

$$a_m \left(\sum_{i \geq 1} d_i u^i\right)^m = a_m \left(\sum_{i \geq 1} d_i u^i\right)^{p^{v_p(m)} s} = a_m \left(\sum_{i \geq 1} d_i^{p^{v_p(m)}} u^{i p^{v_p(m)}}\right)^s.$$

Since $d_{\tilde{B}+1}$ is the first unknown, this affects only u-powers larger than $(\tilde{B}+1)p$. By analyzing the term $a_m \left(\sum_{i \geq 1} d_i^{p^{v_p(m)}} u^{i p^{v_p(m)}}\right)^s$, we get that the smallest term which includes $d_{\tilde{B}+1}$ is of the form

$$a_m s (d_1^{p^{v_p(m)}} u^{p^{v_p(m)}})^{s-1} d_{\tilde{B}+1}^{p^{v_p(m)}} u^{(\tilde{B}+1)p^{v_p(m)}} = a_m s d_1^{(s-1)p^{v_p(m)}} d_{\tilde{B}+1}^{p^{v_p(m)}} u^{(s-1)p^{v_p(m)} + (\tilde{B}+1)p^{v_p(m)}}.$$

Since the roots are distinguished modulo $u^{\nu_{\max}+1}$, by our assumption, we have that

$$(s + \tilde{B})p^{v_p(m)} = \nu_{\max} + 1 \Leftrightarrow \tilde{B} = \frac{\nu_{\max} + 1 - m}{p^{v_p(m)}} \leq \begin{cases} \lceil \frac{\nu_{\max}}{p^{v_p(m)}} \rceil & p \mid m \\ \nu_{\max} + 1 - m & p \nmid m. \end{cases} \quad \square$$

Remark 6.1.48
It is worth pointing out that it is needed to consider $\tilde{\sigma}(u) \mod u^{\nu_{\max}+1}$ *to get a valid permutation. Particularly, although by Remark 6.1.47 we compute* d_i *for every* $1 \leq i \leq \tilde{B}$, *where* $\tilde{B} \leq \nu_{\max}$ *we consider*

$$\tilde{\sigma}(u) = \sum_{i=1}^{\tilde{B}} d_i u^i \mod u^{\nu_{\max}+1},$$

which results in gaining a range in precision for performing computations.

We present the algorithm of the recommended precision.

Algorithm 26: Precision Recommendation [RecommendedPrecision]

```
Input: The list rSF := {$\alpha_1, \ldots, \alpha_n$} of the roots of a polynomial $f$.
Output: The recommended precision $\nu_{\max}$.
Compute $S \leftarrow [v_u(\alpha_1 - \alpha_i) : 2 \leq i \leq |n|]$;
Choose the maximum element in $S$ and denote it by $\nu_{\max}$;
return $\nu_{\max}$;
```

We aim to construct the automorphisms of $\operatorname{Spl}(f)$ over $K := \mathbb{F}_q((t))$ as permutation elements.

We describe the construction of the automorphisms in the general case. Let $K_\ell = \operatorname{Spl}(f) = \mathbb{F}_{q_\ell}((u_\ell))$, with $\ell \geq 1$ be the splitting field of f.

$$\begin{tikzcd}
K_\ell := \mathbb{F}_{q_\ell}((u_\ell)) \arrow[d, dash] \\
K_{\ell-1} := \mathbb{F}_{q_{\ell-1}}((u_{\ell-1})) \arrow[d, dash, dashed] \\
K_2 := \mathbb{F}_{q_2}((u_2)) \arrow[d, dash] \\
K_1 := \mathbb{F}_{q_1}((u_1)) \arrow[d, dash] \\
K := \mathbb{F}_q((t))
\end{tikzcd}$$

Figure 6.1: **Tower of fields of Splitting Field**

We would like to compute $\tau \in \mathrm{Gal}(K_\ell/K)$ such that $\tau(u_i) \neq u_i$ for $1 \leq i \leq \ell-1$. In order to achieve the automorphism τ we perform the following procedure. We factorize the polynomial f into irreducible factors over K. If f splits over K, then $[K_\ell : K] = 1$ and the result is trivial in this case. Thus, we suppose that $[K_\ell : K] > 1$ and let

$$f(X) = h_1(X) \cdots h_l(X) \in K[X]$$

be the factorization of f into irreducible factors. If f is an irreducible polynomial over K, then $h_1 = f$. We know that $\deg(h_1) > 1$, otherwise we cannot have an extension field. Without loss of generality we assume that $h_1 = \min(K_i/K_{i-1})$ and let

$$\{\alpha_{(i)_1}, \dots, \alpha_{(i)_{n_i}}\}$$

be its roots. Then there is a K-automorphism $\tau_i : K_\ell \to K_\ell$ such that $\tau_i(\alpha_{(i)_1}) = \alpha_{(i)_a}$ with $a \neq 1$, as explained above. Nevertheless, in case of a constant field extension at this step, i.e. $\mathbb{F}_{q_i} \neq \mathbb{F}_{q_{i-1}}$, we construct the non-trivial automorphism $\tau_i \in \mathrm{Aut}(K_\ell)$ as defined in Remark 6.1.4.

After the construction of the automorphism we compute the corresponding permutation μ_{τ_i} of the roots of the polynomial and the permutation group G which is generated by μ_{τ_i}. We compute automorphisms until the set of the corresponding permutations $\{\mu_{\tau_i}, \mu_{\tilde{\tau}_i} \dots\}$ of them generate the $\mathrm{Gal}(K_i/K_{i-1})$. We choose non-trivial automorphisms, this means that $\tilde{\tau}_i(\alpha_{(i)_1}) \neq \tau_i(\alpha_{(i)_1})$ and especially the image of $\tilde{\tau}_i(\alpha_{(i)_1})$ is not an element of the orbit $\mathrm{Orbit}_G(\alpha_{(i)_1})$ of the current permutation group G. We repeat the above procedure computing recursively Galois groups

$$\mathrm{Gal}(K_\ell/K_{\ell-1}), \mathrm{Gal}(K_\ell/K_{\ell-2}), \mathrm{Gal}(K_\ell/K_{\ell-3}), \dots, \mathrm{Gal}(K_\ell/K_1).$$

After that we compute automorphisms $\tau \in \mathrm{Gal}(K_\ell/K)$ such that $\tau(u_j) \neq u_j$ for every $1 \leq j \leq \ell$. We repeat this procedure until we finally compute a set S of the corresponding permutations of the automorphisms which generate the whole Galois group of the polynomial. In other words, $G := \mathrm{Gal}(K_\ell/K) = \langle S \rangle$.

We emphasize that for the description above it is important that in every intermediate step we compute distinct automorphisms which fix the ground field of the current relative extension. Under the same conditions and notation as above, we aim to compute $\tau : K_\ell \to K_\ell$ such that $\tau(\alpha_{(i)_1}) = \alpha_{(i)_a}$, with $a \neq 1$ and $\tau(u_{i-1}) = u_{i-1}$.

As we have already explained, we know that there exist as many as $[N : K(\alpha_{(i)_1})]$ different automorphisms of K_ℓ such that $\tau(\alpha_{(i)_1}) = \alpha_{(i)_a}$. However, not all of them have the property to fix the ground field of the relative extension. In order to have access to all of them, we take advantage of the parameter Γ (as studied above, see Remark 6.1.42). Therefore, from a computational point of view, if we properly choose the parameter Γ, then we can select only those which fix u_{i-1}, and so the approach above can run efficiently. To do so, it is a challenging task, and unfortunately, we do not have a clear solution to this matter.

In our algorithm, we follow the above approach but we do not guarantee that in every intermediate step we compute only automorphisms which fix the current ground field. If an automorphism does not fix the current ground field but fixes the base field K, we keep it. Consequently, this can lead to computations of automorphisms that already existed in the Galois group.

Let us now outline the process applied to our algorithm. As we have analyzed, we follow the idea of computing automorphisms based on "top to bottom"- concept. In particular, we consecutively compute automorphisms corresponding to the steps of splitting field tower starting from top to bottom. For every step, we determine the roots of the irreducible polynomial in that relative extension. Then we choose a pair of that roots that belong to different orbits and compute an automorphism σ. Let $\alpha_{i_1}, \alpha_{i_2}$ be two roots of the polynomial at step i. Next we compute a σ satisfying $\alpha_{i_1} \mapsto \alpha_{i_2}$.

Remark 6.1.49
It should be noted that for the computation of $\sigma : \alpha_{i_1} \mapsto \alpha_{i_2}$, we fix the tuple called Γ. This is important as we can construct $[N : K(\alpha_{(i)_1})]$ different automorphisms of K_ℓ such that $\sigma(\alpha_{i_1}) = \alpha_{i_2}$.

We repeat the above process until we reach step one. At step one, we apply our approach, and choose roots that belong to different orbits until we have only one orbit. In that case, we can have two options. In the first one, we have computed the whole Galois group, and so the algorithm terminates. In the second one, though, a part of the Galois group is missing, and so we should repeat computing automorphisms. In that case, we choose the roots α_1 and α_2 in order to compute automorphisms σ mapping α_1 to α_2. The vital part in that point is the choice of the tuple called Γ, (see Remark 6.1.42). The different choices of the aforementioned tuple result in different automorphisms σ such that $\sigma(\alpha_1) = \alpha_2$. We repeat the above computation of σ such that $\sigma(\alpha_1) = \alpha_2$ until we compute the whole Galois group, and therefore the algorithm terminates.

Although we can use the aforementioned approach for Galois group computation, it is worth mentioning that we make some adjustments to this approach when we use the combined approach of splitting field computation. Employing the latter to determine the splitting field, we firstly compute the field T, and so the tower of fields is different than the one described in Figure 6.1. The new tower of fields will be of the following form

$$\begin{tikzcd}
 & & N := K_\ell = \mathbb{F}_{q_\ell}((u_\ell)) \arrow[d, dash, "n_\ell"'] \\
 & & K_{\ell-1} := \mathbb{F}_{q_{\ell-1}}((u_{\ell-1})) \arrow[d, dash, dashed] \\
 & & K_3 := \mathbb{F}_{q_3}((u_3)) \arrow[d, dash] \\
 & & K_2 := \mathbb{F}_{q_2}((u_2)) \arrow[d, dash] \\
 & & KT =: K_1 := \mathbb{F}_{q_1}((u_1)) \arrow[d, dash] \\
K := K[X]/\langle f \rangle \arrow[urr, dash, dashed] \arrow[dr, dash] & & T := \mathbb{F}_{q_0}((u_0)) \arrow[dl, dash] \\
 & K := \mathbb{F}_q((t)) &
\end{tikzcd}$$

Figure 6.2: **Splitting Field: Combined Approach**

Considering the above issue of computing already existing automorphisms and aiming to accelerate the Galois group computation, we present another approach taken to construct the automorphisms.

To begin with, when $[T : K] > 1$ and a constant field extension is involved in T, then we compute the Frobenius automorphism, as defined in Remark 6.1.4, corresponded to T/K.
In case $[T : K] = 1$ and a constant field extension is included in the splitting field computation, then we determine the Frobenius automorphism which corresponds to that case, by Remark 6.1.4.

After that, we compute the automorphisms which correspond to last step. To do so, we employ another method devoted to the construction of the Galois group in the last step. We will examine that thoroughly in a later section.

Thereafter, we compute automorphisms using the orbits of the roots. To further explain, the way choosing the roots depends on the fact that they should be included in different orbits of the already partial computed Galois group. Thus, we repeat the process of choosing a pair of roots provided that they belong to different orbits. Having chosen the roots, let us denote them by α_i, α_j with $i \neq j$ and α_i, α_j belong to different orbits, then we compute the automorphism σ such that maps α_i to α_j, and so we determine $\sigma(u_\ell)$. Applying σ on the roots, we take the corresponding permutation, and then we can compute the Galois group.

Remark 6.1.50
It should be noted that the determination of σ relies on the tuple called Γ, (see Remark 6.1.42). This means that the computation of an automorphism σ depends on the choice of a pair of the roots as well as the tuple Γ.

Comment 6.1.51
For the choice of Γ, we follow the same ideas as we did before in which we construct the automorphisms applying the idea from top to bottom (as described above).

We repeat the above process until we reach the point that the already partial computed Galois group has only one orbit. This can lead to two cases. The first one is that we have already computed the whole Galois group, and therefore the algorithm terminates. The second one concerns the situation in which we do not have the whole Galois group. In such a case, we proceed to the old code with step equals to one.

The main advantage of this method is that we compute many distinct automorphisms which either are enough to determine the whole Galois group, or only a part of the Galois group is missing. This means that only in the latter case is it possible to deal with the issue of computing already existing automorphisms, since the old approach will be used to determine the entire Galois group.

For the sake of completeness and clarity, we summarize the method of computing the automorphisms by using the orbits in an algorithmic form, which is given below.

Algorithm 27: Construct Permutations Using Orbits [ConstructPermutationsUsingOrbits]

```
Input: Given $G$, PG, a list rSF of the roots of the initial polynomial $f$, Spl($f$)
Output: A group $G$, a list PG with the permutations of the roots and the parameters $\Gamma, \Delta$.
Define $O(G) \leftarrow \text{Orbits}(G)$;
Keep only the orbits which includes the roots of the current polynomial. This is meaningful for reducible
  polynomials. Denote them by $O$;
$v_p(\alpha) \leftarrow \gamma_1 \cdots \gamma_l$;
$n(B) \leftarrow \min\{v_p(v_{u_\beta}(\alpha)), n(S)\}$ and $n(B) \leftarrow \begin{cases} n(B)+1 & \text{if } l>1 \text{ or } v_p(v_{u_\beta}(\alpha)) \\ n(B) & \text{else} \end{cases}$;
$\Gamma \leftarrow [1, \dots, 1, 2]$, where $|\Gamma| = n(B)$ and $s \leftarrow 1$;
while $n(O) > 0$ and $n(O(G)) > 1$ do
    $O(G) \leftarrow [x \in O(G) \mid x \in O[1]]$;
    Select $i_0 \leftarrow \begin{cases} O(G)[1][1] & \text{if } s=1 \\ O(G)[n(O(G))-1][1] & \text{else} \end{cases}$ and $j_0 \leftarrow \begin{cases} O(G)[2][1] & \text{if } s=1 \\ O(G)[n(O(G))][1] & \text{else} \end{cases}$;
    $\alpha_{i_0} \leftarrow \alpha_{i_0} \mod u_\beta^{\nu_{\max}}$ and $\alpha_{j_0} \leftarrow \alpha_{j_0} \mod u_\beta^{\nu_{\max}}$;
    if $\alpha_{i_0} = u_\ell$ then im $\leftarrow \alpha_{i_0}$;
    else if $\alpha_{j_0} = u_\ell$ then im $\leftarrow \alpha_{j_0}$;
    else
        Select $\Gamma$ as analyzed above and compute im, $\Delta, \Gamma \leftarrow \text{ConstructImage}(\alpha_{i_0}, \alpha_{j_0}, \text{Spl}(f), v_{\max}, \Gamma, \Delta)$
    Define the homomorphism $h : \text{Spl}(f) \to \text{Spl}(f)$ such that $h(u_\ell) = \text{im}$;
    Apply the homomorphism $h$ to the roots of $f$ in order to construct the corresponding permutation pe
      of the roots. In the case $\mathbb{F}_{\tilde{q}} = \mathbb{F}_q$ perform the following: If $u_\ell = \text{im}$, then set pe $:= (1, 2, \dots, n)$. Else
      compute the elements $h(\rho)$, for every $\rho \in$ rSF with precision $\nu_{\max} + 1$ and construct the
      corresponding permutation pe of the roots;
    Add the resulting permutation in the set PG, PG $=$ PG $\cup \{\text{pe}\}$;
    $G \leftarrow \langle \text{PG} \rangle \leqslant S_n$;
    Define $O(G) \leftarrow \text{Orbits}(G)$;
    Keep only the orbits which includes the roots of the current polynomial. This is meaningful for
      reducible polynomials. Denote them by $O$;
    if $|G| = [\text{Spl}(f) : K]$ then break;
    $s \leftarrow s + 1$;
return $G$, PG, $\Gamma, \Delta$;
```

Moreover, the computation of Frobenius automorphism is summarized in the next algorithm.

Algorithm 28: Automorphism Constant Field Extension [AutomorphismConstantFieldExtension]

```
Input: Given $K := \mathbb{F}_q((t))$, the list subst of the embedded elements, a list rSF of the roots of the initial
       polynomial $f$, $\mathrm{Spl}(f) = \mathbb{F}_{q_\ell}((u_\ell))$, $\nu_{\max}, n, i_0, j_0$
Output: A group $G$, a list PG with the permutations of the roots.
Define the homomorphism $\sigma : \mathrm{Spl}(f) \to \mathrm{Spl}(f)$ such that $\sigma(u_\ell) = \mathrm{im}$, as defined in Remark 6.1.4 by using
  $i_0$ and $j_0$ in order to determine in which step the constant field extension occurs, and $\sigma(w) = w^q$, where
  $w$ is the generating element of $\mathbb{F}_{q_\ell}$;
Apply the homomorphism $\sigma$ to the roots of $f$ to construct the corresponding permutation $\mu_\sigma$ of the roots;
Add the resulting permutation in the set PG, PG $\leftarrow$ PG $\cup \{\mu_\sigma\}$;
$G \leftarrow \langle \mathrm{PG} \rangle \leqslant S_n$;
return $G$, PG;
```

It is worth pointing out that in the last step we can compute its Galois group using another approach which is analyzed in the next section.

6.2 Galois Groups in the Last Step

For the computation of automorphisms in the last step in case of a totally ramified extension, i.e automorphisms of $\mathrm{Gal}(K_\ell/K_{\ell-1})$, we can take advantage of using an extension field of $K_{\ell-1}$. Let us now describe this approach.

We begin by recalling our notation. Let $f \in K[X]$ be a polynomial and its tower of fields $K := \mathbb{F}_q((t)) \leqslant K_1 \leqslant \cdots \leqslant K_\ell := \mathbb{F}_{q_\ell}((u_\ell)) := \mathrm{Spl}(f)$ which is obtained by the splitting field computation. In what follows, $n_\ell := [K_\ell : K_{\ell-1}]$ and $L_\ell := K_{\ell-1}[X]/\langle f_\ell \rangle$ stands for the stem field of the irreducible factor f_ℓ of f, and so $L_\ell = K_{\ell-1}(\alpha)$ such that $f_\ell(\alpha) = 0$. Additionally, $M_\ell := K_{\ell-1}[X]/\langle g_\ell \rangle$ denotes the stem field of the Eisenstein polynomial g_ℓ such that $L_\ell \cong M_\ell$. Also, we write π_ℓ for the primitive element of M_ℓ, that is $M_\ell := K_{\ell-1}(\pi_\ell)$ so that $g_\ell(\pi_\ell) = 0$. In particular, we know that $\pi_\ell := \sum_{i=0}^{n_\ell - 1} \gamma_i \alpha^i$ with $\gamma_i \in K_{\ell-1}$.

We emphasize that we have access to the aforementioned field M_ℓ throughout our approach for the splitting field computation.

$$\begin{tikzcd}[arrows=dash]
K_\ell := \mathbb{F}_{q_\ell}((u_\ell)) \arrow[r, phantom, "\cong"] \arrow[d, "n_\ell"'] & L_\ell := K_{\ell-1}[X]/\langle f_\ell\rangle \arrow[r, phantom, "\cong"] \arrow[dl] & M_\ell := K_{\ell-1}[X]/\langle g_\ell\rangle \arrow[dll] \\
K_{\ell-1} := \mathbb{F}_{q_{\ell-1}}((u_{\ell-1})) \arrow[d, dashed] & & \\
K_2 := \mathbb{F}_{q_2}((u_2)) \arrow[d] & & \\
K_1 := \mathbb{F}_{q_1}((u_1)) \arrow[d] & & \\
T := \mathbb{F}_{q_0}((u_0)) \arrow[d] & & \\
K := \mathbb{F}_q((t)) & &
\end{tikzcd}$$

According to the above diagram the computation of $\mathrm{Gal}(K_\ell/K_{\ell-1})$ is reduced to the computation of $\mathrm{Gal}(M_\ell/K_{\ell-1})$. The advantage of computing the latter lies in the fact that the structure of the extension $M_\ell/K_{\ell-1}$ is computationally more efficient for determining the automorphisms.

For the computation of $\mathrm{Gal}(M_\ell/K_{\ell-1})$ we need to have access to the roots of g_ℓ. To do so, obviously, we can factorize g_ℓ over $K_{\ell-1}$. Nevertheless, this factorization can be time-consuming. We recall that in the "Improved Approach" of splitting field computation, we make use of the corresponding Eisenstein polynomials for generating the relative extension in every step. Thus, we have already computed the roots of the polynomial g_ℓ, and so we just retrieve them from the splitting field computation and denote them by $R(g_\ell)$. We point out that for every $\rho \in R(g_\ell)$ it holds that $\rho \in K_\ell$. By using the map

$$\varphi : K_\ell \to M_\ell,\ u_\ell \mapsto \pi_\ell,$$

we get the roots of g_ℓ in M_ℓ, that is $R(g_\ell) = [\varphi(\rho)) \ : \ \rho \in R(g_\ell)]$.

After that we consider the roots of f to be elements of M_ℓ, and denote them by $\tilde{R}(f)$. Then we apply the normal procedure determining the automorphisms. For the sake of completeness of the above procedure, we will give it in algorithmic form below.

The only point remaining concerns the behavior of the precision. Particularly, according to our method, we increase the precision in every consecutive step of the splitting field construction. However, we do not need that precision when working over the extension field M_ℓ, and hence its reduction plays a vital part throughout the computation of $\mathrm{Gal}(M_\ell/K_{\ell-1})$. The key point is to check the roots with M_ℓ-precision $\nu_{\max}+1$. This is why we reduce the $u_{\ell-1}$-precision of $K_{\ell-1}$ to be equal to $\frac{\nu_{\max}}{n_\ell}+1$.

The above procedure is summarized in an algorithmic form, which is given below.

Algorithm 29: Galois Group: Last Step [GaloisGroupLastStepExt]

```
Input: A list subst of the substitutions that we make of f in order to compute the
       splitting field, a list RootsCurPoly of the roots of the current irreducible
       polynomial, the list $R(g_\ell)$ of the roots of $g_\ell$, a list Roots_orig of the roots of the
       initial polynomial $f$ expressed in the original step, the order $n_\ell$ of the group and a
       dataset data_flds of the elements $\langle K_i, f_i, g_i, f(K_i/K), M_i\rangle$ for $1 \le i \le \ell$.
Output: A group $G_l$ and a group $PG_l$ with the permutations of the roots.
Initialize $n \leftarrow |\text{Roots\_orig}|$, $G_l \leftarrow \langle \mathrm{id}\rangle \leqslant S_n$ and $\mathrm{PG}_l \leftarrow [\,]$;
Set $\ell$ to be the last step and retrieve from the list data_flds the extension field
  $M_\ell := K_{\ell-1}[X]/\langle g_\ell\rangle = K_{\ell-1}(\pi_\ell)$, its primitive element $\pi_\ell$, its generating polynomial $g_\ell$
  and the corresponding irreducible factor $f_\ell$ of the initial polynomial $f$, whose roots are
  $R(f_\ell) :=$ RootsCurPoly;
Define Roots_extra to be the list of the additional roots found in last step $\ell$ and
  $n_e \leftarrow |\text{Roots\_extra}|$;
Set the precision of $M_\ell$ to be equal to prec_new $:= n_\ell \cdot \nu_{\max}$ ;
Determine the roots of $g_\ell$, $R(g_\ell) = [\varphi(\rho)) \ : \ \rho \in R(g_\ell)]$ ;
Modify the roots of $f$ to be elements of $M_\ell$ and denote them by $\tilde{R}(f)$. Particularly,
  regarding the roots of $f$ found up to step $\ell - 1$, we consider them as Laurent series
  elements of $K_{\ell-1}$ by using the embedded elements of the list subst. Concerning the roots
  of $f$ found in last step $\ell$, we transform them as elements of $M_\ell$ by using the map $\varphi$. That
  is,
        $\tilde{R}(f) \leftarrow [\varphi(\varrho) \mod u_\ell^z \ : \ \varrho \in \text{Roots\_orig}]$,
  where $z := \frac{\nu_{\max}}{n_\ell} + 1$;
Initialize $a \leftarrow 1$;
repeat
    Define the automorphism $\tau : M_\ell \to M_\ell,\ \pi_\ell \mapsto \rho,$ where $\rho := R(g_\ell)[a]$ ;
    Apply the homomorphism $\tau$ to the modified roots $\tilde{R}(f)$ of $f$ in order to construct the
      corresponding permutation $\mu_\tau$ of the roots;
    Add the resulting permutation in the set $\mathrm{PG}_l$, $\mathrm{PG}_l = \mathrm{PG}_l \cup \{\mu_\tau\}$;
    Define $G_l \leftarrow \langle \mathrm{PG}_l\rangle \leqslant S_n$ ;
    Set $a \leftarrow a + 1$;
until $|G_l| \ge n_\ell$;
return $G_l, \mathrm{PG}_l$;
```

Having analyzed this special case concerning the automorphisms in the last step, we can now return to compute an automorphism in the general case.

In the algorithm for the computation of the permutations we should determine the roots of an irreducible factor of the given polynomial. For the original polynomial we know the roots by the splitting field computation. So we should check which of the roots are the roots of the current irreducible factor. We perform this procedure with a separate function which we call it "RootsOfCurrentPolynomial".

Now we can give the algorithm for computing permutations of the roots of a polynomial f which correspond to automorphisms of its splitting field. We establish some notation used throughout the algorithm. Firstly, $\mathrm{Spl}(f) := \mathbb{F}_{q_\ell}((u_\ell))$ stands for the splitting field of f over K and rSF denotes a list of the roots of f. Also, $\mathrm{subst} = \{\Phi_1(t) \in L_1, \Phi_{j+1}(u_j) \in L_{j+1}, \text{ with } 1 \leq j \leq \ell - 1\}$ denotes the list of the substitutions that we make of f in order to compute the splitting field, $T := \mathbb{F}_{q_{i_0}}((u_{i_0}))$ and data is a dataset with the minimal polynomial in each intermediate field extension.

Note that in our algorithm we define $T := \mathbb{F}_{q_{i_0}}((u_{i_0}))$ where $i_0 = 1$ or 2. The case $i_0 = 1$ occurs when $e(T/K) > 1$ or $\mathfrak{f}(T/K) > 1$ and $i_0 = 2$ when $e(T/K) > 1$ and $\mathfrak{f}(T/K) > 1$. Thus, the number of steps needed for the splitting field computation is $\ell - i_0$, according to our notation.

Algorithm 30: Automorphisms As Permutations, [AutomorphismsAsPermutations]

```
Input: A polynomial $f$ of degree $n$ over $K = \mathbb{F}_q((t))$.
Output: A list with the permutations of the roots, the order of the Galois group, the
        splitting field and the roots of the polynomial.
Apply the splitting field algorithm,
    $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data}, N, T, [T : K], \mathrm{data\_flds}, \mathrm{Roots\_orig}, \mathrm{RootsASubstStep},$
        $R(g_d), \mathrm{IrredFactors}, T_e \leftarrow \mathrm{SplittingField\_final}(f)$
  Apply Algorithm 26 to choose the needed size, $\nu_{\max} \leftarrow \mathrm{RecommendedPrecision(rSF)}$;
Initialize PG $\leftarrow$ [ ], WSizeG $\leftarrow$ 1 and $G \leftarrow \langle \mathrm{id} \rangle \leqslant S_n$;
while $|G| <$ size do
    for $s = \ell$ to 1 by step -1 do
        Define $f_s \leftarrow \min(u_s, K_{s-1})$, $n_s \leftarrow \deg(f_s)$, $n_2 \leftarrow \deg(h_s)$, where $h_s$ is the
          corresponding irreducible factor of $f$ at the step $s$, and compute $R$ to be the list of
          the roots of $f_s$, by
              $\mathrm{RootsCurPol} \leftarrow \mathrm{RootsOfCurrentPolynomial}(f_s, \mathrm{rSF}, \mathrm{subst}, s)$.
          if $\mathbb{F}_{q^{s-1}} = \mathbb{F}_{q^s}$, then WSizeG $\leftarrow$ WSizeG $\cdot\, n_s$ else WSizeG $\leftarrow$ WSizeG $\cdot\, \frac{n_s}{n_2}$;
        Check if the variable $u_\ell$ of the $\mathrm{Spl}(f)$ is a root of the polynomial $f$ and keep its
          position in the roots, that is $\mathrm{log\_value}, \mathrm{pos} := \mathrm{Check(RootsCurPol, Spl(f))}$;
        Initialize $a \leftarrow 1$, par $\leftarrow 1$ and $n_1 \leftarrow |\mathrm{RootsCurPol}|$;
        repeat
            if $\mathbb{F}_{q^{s-1}} = \mathbb{F}_{q^s}$ then
                if the variable $u_\ell$ of the $\mathrm{Spl}(f)$ is a root of $f$ then
                    Choose a random root $\rho$ of the current polynomial $f_s$ provided that
                      $\rho \notin \mathrm{Orbit}_G(1)$ and define im $\leftarrow \rho$
                else
                    if $|\mathrm{Orbit}_G(1)| = n_1$ then par $\leftarrow$ par + 1 and $a \leftarrow 2$ else Choose $a$ be a
                      random number of the set $\{1, \dots, n_1\}$ such that $a \notin \mathrm{Orbit}_G(1)$;
                    Apply Algorithm 25 to choose the value of im, that is
                      $\mathrm{im} \leftarrow \mathrm{ConstructImage(RootsCurPol[1], RootsCurPol[a]}, \mathrm{Spl}(f), \nu_{\max}, \Gamma)$
                Define the homomorphism $\sigma : \mathrm{Spl}(f) \to \mathrm{Spl}(f)$ such that $\sigma(u_\ell) = \mathrm{im}$;
            else if $\mathbb{F}_{q^{s-1}} \neq \mathbb{F}_{q^s}$ then
                Define the homomorphism $\sigma : \mathrm{Spl}(f) \to \mathrm{Spl}(f)$ such that $\sigma(u_\ell) = \mathrm{im}$, as
                  defined in Remark 6.1.4, and $\sigma(w) = w^q$, where $w$ is the generating element
                  of $\mathbb{F}_{\tilde{q}}$;
            Apply $\sigma$ to the roots of $f$ to construct the corresponding permutation $\mu_\sigma$ of the
              roots. In the case $\mathbb{F}_{q^{s-1}} = \mathbb{F}_{q^s}$, if $u_\ell = \mathrm{im}$, then set $\mu_\sigma := (1, 2, \dots, n)$;
            Add the resulting permutation in the set PG, PG = PG $\cup \{\mu_\sigma\}$;
            Define $G \leftarrow \langle \mathrm{PG} \rangle \leqslant S_n$ and set a $\leftarrow$ a + 1;
        until $|G| \geq$ WSizeG;
        if $|G| =$ size then break the for-loop;
return PG, size, $\mathrm{Spl}(f)$, rSF
```

Now we give the algorithm which corresponds to the latest version of splitting field computation, which is "Improved Approach"-Using Eisenstein polynomials in every step.

Algorithm 31: Automorphisms As Permutations, [AutomorphismsAsPermutations]

```
Input: A polynomial $f$ of degree $n$ over $K = \mathbb{F}_q((t))$.
Output: A list with the permutations of the roots, the order of the Galois group, the splitting field and
        the roots of the polynomial.
Apply the splitting field algorithm,
    $\mathrm{Spl}(f), \mathrm{size}, \mathrm{rSF}, \mathrm{subst}, \mathrm{data}, N, T, [T:K], \mathrm{data\_flds}, \mathrm{Roots\_orig}, \mathrm{RootsASubstStep},$
        $R(g_d), \mathrm{IrredFactors}, T_e \leftarrow \mathrm{SplittingField\_final}(f)$
Apply Algorithm 26 to choose the needed size, $\nu_{\max} \leftarrow \mathrm{RecommendedPrecision}(\mathrm{rSF})$;
Define $\mathrm{PGL} \leftarrow [\,]$, $\ell \leftarrow |\mathrm{data}|$, $f_\ell \leftarrow \min(u_\ell, \mathbb{F}_{q^{\ell-1}}((u_{\ell-1})))$, $n_\ell \leftarrow \deg(f_\ell)$, and $G \leftarrow \langle \mathrm{id} \rangle \leqslant S_n$ ;
if $[T:K] > 1$ and $\mathbb{F}_{q_{i_0}} \neq \mathbb{F}_q$ then
    Compute the Frobenius automorphism by applying Algorithm 28,
      $G_0, \mathrm{PGL}_0 \leftarrow \mathrm{AutomorphismConstantFieldExtension}(\mathrm{subst}, \mathrm{rSF}, c_0\nu_{\max}, n, \mathrm{size}, i_0, i_0)$ ;
    $\mathrm{PG} \leftarrow \mathrm{PG} \cup \mathrm{PGL}_0$, $\mathrm{WSizeG} \leftarrow |G_0|$ and $\mathrm{CSizeG} \leftarrow |G_0|$;
else if $\mathbb{F}_{q_{i_0}} = \mathbb{F}_q$ and $\mathbb{F}_{q_{i_0}} \neq \mathbb{F}_{q_\ell}$ then
    Determine $j_0$ to be the step in which the constant field extension occurs;
    Compute the Frobenius automorphism by applying Algorithm 28,
      $G_1, \mathrm{PGL}_1 \leftarrow \mathrm{AutomorphismConstantFieldExtension}(\mathrm{subst}, \mathrm{rSF}, \nu_{\max}, n, \mathrm{size}, i_0, j_0)$ ;
    $\mathrm{PG} \leftarrow \mathrm{PG} \cup \mathrm{PGL}_1$, $\mathrm{WSizeG} \leftarrow |G_1|$ and $\mathrm{CSizeG} \leftarrow |G_1|$;
else $\mathrm{PG} \leftarrow \mathrm{PG} \cup \mathrm{PGL}$, $\mathrm{WSizeG} \leftarrow 1$ and $\mathrm{CSizeG} \leftarrow |G|$ ;
$G \leftarrow \langle \mathrm{PG} \rangle \leqslant S_n$;
if $n \neq n_\ell$ then
    Compute $R$ to be the list of the roots of $f_\ell$;
    Compute the Galois group in the last step, i.e $\mathrm{Gal}(K_\ell/K_{\ell-1})$,
      $G_1, \mathrm{PGL} \leftarrow \mathrm{GaloisGroupLastStepExt}(\mathrm{subst}, R, R(g_\ell), \mathrm{Roots\_orig}, n_\ell, \mathrm{data\_flds}, \mathrm{size}, \nu_{\max})$;
    $\mathrm{PG} \leftarrow \mathrm{PG} \cup \mathrm{PGL}$, $G \leftarrow \langle \mathrm{PG} \rangle \leqslant S_n$ and $\mathrm{CSizeG} \leftarrow |G|$;
    $\mathrm{lastStep} \leftarrow \begin{cases} \ell & \text{if } [T:K] = 1 \\ \ell - 1 & \text{else} \end{cases}$ and $\mathrm{startStep} \leftarrow \mathrm{lastStep} - 1$
else $\mathrm{startStep} \leftarrow \begin{cases} \ell & \text{if } [T:K] = 1 \\ \ell - 1 & \text{else} \end{cases}$ ;
if $f$ is irreducible then $\mathrm{validOrbitsRoots} \leftarrow [[1 \ldots n]]$;
else Group the roots of each irreducible factor of $f$ into lists and denote it by validOrbitsRoots;
$G, \mathrm{PG}, \Gamma, \Delta \leftarrow$
  $\mathrm{ConstructPermutationsUsingOrbits}(G, \mathrm{size}, \mathrm{PG}, \mathrm{rSF}, \mathrm{Spl}(f), \nu_{\max}, n(S), v_{u_\ell}(\alpha), \mathrm{validOrbitsRoots})$;
$\mathrm{CSizeG} \leftarrow |G|$, $\mathrm{startStep} \leftarrow 1$ and $a_o \leftarrow 2$;
while $|G| < \mathrm{size}$ do
    for $s = \mathrm{startStep}$ to 1 by step -1 do
        $\tilde{s} \leftarrow \begin{cases} s & \text{if } [T:k] = 1 \\ s + i_0 - 1 & \text{else} \end{cases}$ ;
        Define $f_{\tilde{s}} \leftarrow \min(u_{\tilde{s}}, K_{\tilde{s}-1})$, $n_{\tilde{s}} \leftarrow \deg(f_{\tilde{s}})$, $n_2 \leftarrow \deg(h_{\tilde{s}})$, where $h_{\tilde{s}}$ is the corresponding
          irreducible factor of $f$ at step $s$;
        if $\mathbb{F}_{q^{s-1}} = \mathbb{F}_{q^s}$ then $\mathrm{WSizeG} \leftarrow \mathrm{WSizeG} \cdot n_s$ else $\mathrm{WSizeG} \leftarrow \mathrm{WSizeG} \cdot \frac{n_s}{n_2}$;
        Compute $R$ to be the list of the roots of $h_{\tilde{s}}$;
        if $u_\ell$ is a root of $f$ then Assume $\alpha_1 = u_\ell$;
        Initialize $a \leftarrow 1$, $\mathrm{par} \leftarrow 1$ and $n_1 \leftarrow |R|$;
        repeat
            if $\mathbb{F}_{q^{s-1}} = \mathbb{F}_{q^s}$ then
                if $u_\ell$ is a root of $f$ then Choose a random root $\rho$ of the current polynomial $h_{\tilde{s}}$ provided
                  that $\rho \notin \mathrm{Orbit}_G(1)$ and define $\mathrm{im} \leftarrow \rho$ ;
                else
                    if $|\mathrm{Orbit}_G(1)| = n_1$ then $\mathrm{par} \leftarrow \mathrm{par} + 1$ and $a \leftarrow 2$ and select $\Gamma$;
                    else Choose $a$ be a number of the set $\{1, \ldots, n_1\}$ such that $a \notin \mathrm{Orbit}_G(1)$;
                    if $\alpha_1 = u_\beta$ then $\mathrm{im} \leftarrow \alpha_1$;
                    else if $\alpha_a = u_\beta$ then $\mathrm{im} \leftarrow \alpha_a$;
                    else
                        Define $c_0 \leftarrow \begin{cases} 1 & \text{if } f \text{ is irreducible} \\ n_1 & \text{else} \end{cases}$ ;
                        Compute $\mathrm{im}, \Delta, \Gamma \leftarrow \mathrm{ConstructImage}(\alpha_1, \alpha_a, \mathrm{Spl}(f), c_0 \cdot \nu_{\max}, \Gamma, \Delta)$
                Define the homomorphism $\sigma : \mathrm{Spl}(f) \to \mathrm{Spl}(f)$ such that $\sigma(u_\ell) = \mathrm{im}$;
            else if $\mathbb{F}_{q^{s-1}} \neq \mathbb{F}_{q^s}$ then
                Define the homomorphism $\sigma : \mathrm{Spl}(f) \to \mathrm{Spl}(f)$ such that $\sigma(u_\ell) = \mathrm{im}$, as defined in Remark
                  6.1.4, and $\sigma(w) = w^q$, where $w$ is the generating element of $\mathbb{F}_{q^s}$;
            Apply $\sigma$ to the roots of $f$ to construct the corresponding permutation $\mu_\sigma$ of the roots. In the
              case $\mathbb{F}_{q^{s-1}} = \mathbb{F}_{q^s}$, if $u_\beta = \mathrm{im}$, then set $\mu_\sigma := (1, 2, \ldots, n)$;
            $\mathrm{PG} \leftarrow \mathrm{PG} \cup \{\mu_\sigma\}$, and define $G \leftarrow \langle \mathrm{PG} \rangle \leqslant S_n$ and set $\mathrm{a} \leftarrow \mathrm{a} + 1$;
        until $|G| \geq \mathrm{WSizeG}$;
        if $|G| = \mathrm{size}$ then break the for-loop;
        $a_o \leftarrow a$;
return PG, size, $\mathrm{Spl}(f)$, rSF;
```

6.3 Galois Groups

Let $f \in K[X]$ be a polynomial of degree n. The Galois group of f over K can be embedded in the symmetric group S_n. We can determine each automorphism $\sigma \in \mathrm{Gal}(\mathrm{Spl}(f)/K)$ associating its permutation of the roots which in turn can be viewed as a permutation on the subscripts of the roots $\alpha_1, \dots, \alpha_n$. So this computation depends on the choice of ordering the roots of f. Numbering the roots of f in different ways can identify the Galois group of f with different subgroups of the symmetric group S_n. Two different choices for ordering the roots of f can lead to different subgroups of S_n, but those subgroups will be conjugate subgroups by a permutation of S_n.

Example 6.3.1
Let $f(X) = X^5 + tX + t \in \mathbb{F}_5(t)[X]$, which is a strongly Eisenstein polynomial. The polynomial has five roots α_1, α_2, α_3, α_4, α_5 and the splitting field of f is $\mathrm{Spl}(f) = \mathbb{F}_5((u_2))$. According to that labelling of the roots we have that the Galois group of $f(X)$ is a permutation group acting on a set of cardinality 5 of order 20 with generators $(1,4)(2,3)$, $(1,3)(4,5)$, $(1,2,5,4)$.

If we fix another labelling of the roots, then we have that the Galois group of f is a permutation group acting on a set of cardinality 5 of order 20 with generators $(2,5,4,3)$, $(1,2,5,3)$.

But those two subgroups of S_5 are conjugate by the permutation $(4, 5)$.

Thus, in general the Galois group of f over K does not have a canonical embedding in S_n. However, we identify the image of the Galois group in S_n up to a conjugation by a permutation.

Now we are able to present the algorithm of the Galois group computation.

Algorithm 32: Galois Group Computation [GaloisGroup(f)]

```
Input: A polynomial $f$ over $K$.
Output: The Galois group of the polynomial, a list of the roots of the polynomial and the splitting field of it.
Apply the permutation Algorithm 31 on $f$,
        $\mathrm{PerElm}, \mathrm{Ord}, \mathrm{Spl}(f), \mathrm{rSF} \leftarrow \mathrm{AutomorphismsAsPermutations}(f)$;
if $\mathrm{Ord} = |S_n|$, where $n = \deg(f)$ then
    define the group $G := S_n$ and return $G$, rSF, Spl($f$);
else
    Define $G$ to be the subgroup of $S_n$ which is generated by the set PerElm, and return $G$, rSF, Spl($f$);
```

In conclusion, in our approach, the Galois group is presented as a permutation group of the roots. We aim at the computation of the Galois group in a direct way. According to the description, we can see that this method depends on the splitting field computation. Therefore, this method uses the representation of the successive fields of the tower as the Laurent series fields $\mathbb{F}_{q_i}((u_i))$. Moreover, we explicitly present every root of the given polynomial as an element of the splitting field $\mathbb{F}_{q_\ell}((u_\ell))$. Afterwards, we present the Galois group as a permutation group of them.

According to Romano [41], we already know the Galois group of strongly Eisenstein polynomials by the following theorem.

Theorem 6.3.2
Let $f \in K[X]$ be a strongly Eisenstein polynomial of degree p^d, then the Galois group of f is

$$\mathrm{Gal}(f) \cong C_p^d \rtimes C_{p^d-1} \rtimes C_d.$$

We use this theorem in order to check the correctness of our algorithms.

6.4 Examples

In general, it is possible to compute the Galois group of any separable polynomial f given over some local function field K. There are no further restrictions on f, e.g. f can be reducible or it is not necessary that f is Eisenstein.

The tables in Section A.3 show that it is possible to compute Galois groups of order more than one thousand. We also note that the running times are depending on the valuations of the roots of the given polynomial. We mentioned that Greve's method has been efficient for situations with ramification polygons consisting of less than two segments. In the tables we give several examples with three segments and even some with four segments. Note that the length of the list in the last column give the number of slopes.

Appendix A

Examples

We will list tables of examples. Let us first describe the notation needed to understand them. Let $f \in K[X]$, of degree $\deg(f) = n$, where $K := \mathbb{F}_q((t))$.

- $\mathrm{Spl}(f) = \mathbb{F}_{q^{l_1}}((u_{r_1}))$: It is based on the Splitting Field Approach: A Variation of Basic Approach (Ver1)-Using original factors of f, which is Algorithm 9. Instead of giving $\mathrm{Spl}(f)$, we return l_1 and the number r_1 of field extensions in our tower.
- $\mathrm{Prec}(u_{r_1})$: The needed u_{r_1}-precision for the algorithm based on Ver1.
- T1(sec): The time needed for the splitting field computation based on Ver1.
- $\mathrm{Spl}(f) = \mathbb{F}_{q^{l_2}}((u_{r_2}))$: It based on the Splitting Field Approach: Combined Version (Ver2)-Using original factors of f, which is Algorithm 22. Instead of giving $\mathrm{Spl}(f)$, we return l_2 and the number r_2 of field extensions in our tower.
- $\mathrm{Prec}(u_{r_2})$: The needed u_{r_2}-precision for the algorithm based on Ver2.
- T2(sec): The time needed for the splitting field computation based on Ver2.
- $\mathrm{Spl}(f) = \mathbb{F}_{q^{l}}((u_{r}))$: It based on the Splitting Field Approach: Improved Approach (Ver3)-Using Eisesntein polys, which is Algorithm 22 after applying the idea that in every step we take the corresponding Eisenstein polynomial. Instead of giving $\mathrm{Spl}(f)$, we return l and the number r of field extensions in our tower.
- $\mathrm{Prec}(u_r)$: The needed u_r-precision for the algorithm based on Ver3.
- T: The field $T := \mathbb{F}_{q^{l_T}}((u_{r_T}))$. We return the tuple $(l_T, r_T, t = u_{r_T}^{e(T/K)})$. We also define $u_0 := t$.
- $[T : K] = \mathfrak{f}(T/K)e(T/K)$: The degree of the extension T/K.
- T3(sec): The time needed for the splitting field computation based on Ver3.
- T4(sec): The time needed for the Galois group (including the splitting field) computation.
- $\#\sigma$: The number of computed automorphisms for determining the Galois group.
- $v_{u_r}(\alpha_i - \alpha)$: The set of the valuation of the differences of the roots. In particular,
  $v_{u_r}(\alpha_i - \alpha) := \{v_{u_r}(\alpha_i - \alpha) \;:\; 1 \leq i \leq n\}$
  $v_{u_r}(\alpha_i - \alpha) = v_{u_{r_1}}(\alpha_i - \alpha) = v_{u_{r_2}}(\alpha_i - \alpha)$.
- Slopes: The absolute value of the slopes of the ramification polygon of f.

The computations we give timings for in this appendix we run on an Intel(R) Xeon(R) Gold 6346 CPU 3.10GHz (755 GB RAM) using Magma V2.27-6. Moreover, we remark that the polynomials used in the tables are chosen randomly concerning the situation described at the head of each table.

A.1 Splitting Fields: Comparison of Version 1 to Version 2

In this section we give a table of examples in which we compare the two approaches-"A Variation of the Basic Approach" (Ver1), and "Combined Approach" (Ver2),- for the splitting field computation.

Table A.1: Eisenstein polynomials with more than one segments over $\mathbb{F}_2((t))$ - Ver1 to Ver2

f	**|Spl(f)|**			**Prec**	**[T:K]**	**T**	**T2**			**Prec**	**T1**		**Slopes**
	$= \vert\mathbb{F}_{q^{l_2}}((u_{r_2}))\vert$	l_2	r_2	(u_{r_2})	$= \mathfrak{f}(T/K)e(T/K)$	$(l_T, r_T, t = u_{r_T}^e)$	(sec)	l_1	$\mathbf{r}_1$	(u_{r_1})	(sec)	$v_{u_r}(\alpha_i - \alpha)$	
$x^4 + (t + t^2)x^2 + t^2x + t$	$4 = 2^2$	1	1	230	1	$(1, 0, t = t)$	0.0	1	1	90	0.0	$\{2^2, 4, \infty\}$	$[3, 1]$
$x^4 + t^2x^2 + t^8x + t$	$8 = 2^3$	1	2	358	1	$(1, 0, t = t)$	0.0	1	2	668	0.1	$\{8^2, 48, \infty\}$	$[23, 3]$
$x^4 + t^8x^3 + t^4x^2 + t^{14}x + t$	$8 = 2^3$	1	2	776	1	$(1, 0, t = t)$	0.2	1	2	818	0.4	$\{16^2, 36, \infty\}$	$[17, 7]$
$x^6 + tx + t$	$24 = 2^3 \cdot 3$	2	4	305	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.0	2	3	100	0.0	$\{2^4, 4, \infty\}$	$[1, 0]$
$x^8 + tx^7 + t^2x^5 + t^3x + t$	$24 = 2^3 \cdot 3$	3	2	238	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	0.0	3	2	130	0.0	$\{2^7, \infty\}$	$[1]$
$x^{12} + t^2x^9 + t^5x + t$	$24 = 2^3 \cdot 3$	2	3	503	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.1	2	2	623	0.2	$\{1^8, 8^3, \infty\}$	$[7, 0]$
$x^8 + t^3x^7 + t^2x^6 + tx^4 + t$	$32 = 2^5$	2	3	260	1	$(1, 0, t = t)$	0.1	2	3	364	0.2	$\{4^4, 12^2, 20, \infty\}$	$[9, 5, 1]$
$x^8 + t^3x^7 + t^2x^6 + t^2x^5 + tx^4 + t$	$32 = 2^5$	2	3	474	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.1	2	3	224	0.1	$\{4^4, 8^3, \infty\}$	$[3, 1]$
$x^6 + t^3x^5 + t^2x^3 + t^2x + t$	$48 = 2^4 \cdot 3$	2	5	411	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.2	2	4	240	0.1	$\{4^4, 32, \infty\}$	$[7, 0]$
$x^8 + t^9x^7 + t^4x^6 + t^5x^4 + t$	$64 = 2^6$	2	4	911	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.9	2	4	2243	2.1	$\{24^6, 168, \infty\}$	$[41, 5]$
$x^8 + t^7x^7 + t^2x^6 + tx^4 + t$	$64 = 2^6$	2	4	825	1	$(1, 0, t = t)$	1.3	2	4	1861	1.7	$\{8^4, 24^2, 168, \infty\}$	$[41, 5, 1]$
$x^8 + t^9x^7 + t^2x^6 + tx^4 + t$	$64 = 2^6$	2	4	1215	1	$(1, 0, t = t)$	2.7	2	4	2595	3.3	$\{8^4, 24^2, 232, \infty\}$	$[57, 5, 1]$
$x^8 + t^9x^7 + t^4x^6 + t^3x^4 + t$	$96 = 2^5 \cdot 3$	3	4	911	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	0.9	3	4	2243	2.9	$\{24^6, 168, \infty\}$	$[41, 5]$
$x^{18} + tx^9 + t^2x + t$	$108 = 2^2 \cdot 3^3$	6	3	819	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	0.6	6	2	410	0.5	$\{1^{16}, 10, \infty\}$	$[9, 0]$
$x^{18} + tx^9 + t^3x^3 + t^3x + t$	$108 = 2^2 \cdot 3^3$	6	3	819	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	0.6	6	2	410	0.5	$\{1^{16}, 10, \infty\}$	$[9, 0]$
$x^8 + tx^7 + t^2x^5 + tx^3 + t$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	407	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.1	3	3	139	0.1	$\{10^7, \infty\}$	$[3/7]$
$x^8 + tx^7 + tx + t$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	393	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.0	3	3	123	0.1	$\{8^7, \infty\}$	$[1/7]$
$x^8 + t^2x^6 + tx^4 + t^4x^2 + t^2x + t$	$192 = 2^6 \cdot 3$	2	4	672	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.1	2	4	254	0.1	$\{24^4, 32^3, \infty\}$	$[5/3, 1]$
$x^{18} + t^3x^{11} + t^3x^8 + t^2x^3 + t^5x + t$	$216 = 2^3 \cdot 3^3$	6	4	1779	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	4.2	6	3	1427	3.7	$\{2^{16}, 44, \infty\}$	$[21, 0]$
$x^{10} + t^2x^8 + tx^3 + t$	$320 = 2^6 \cdot 5$	4	6	517	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	0.3	4	5	260	0.2	$\{8^8, 32, \infty\}$	$[3, 0]$
$x^{14} + tx + t$	$336 = 2^4 \cdot 3 \cdot 7$	6	6	547	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.5	6	5	164	0.2	$\{4^{12}, 8, \infty\}$	$[1, 0]$
$x^8 + t^{19}x^7 + t^{25}x^6 + t^{12}x^4 + t^8x^2 + t$	$384 = 2^7 \cdot 3$	2	6	11391	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	232.0	2	5	33253	4413.0	$\{256^6, 2256, \infty\}$	$[93, 29/3]$
$x^8 + t^{19}x^7 + t^{25}x^6 + t^{10}x^4 + t^8x^2 + t$	$384 = 2^7 \cdot 3$	2	6	11391	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	245.5	2	5	33253	4645.4	$\{256^6, 2256, \infty\}$	$[93, 29/3]$
$x^8 + tx^7 + tx^2 + t$	$384 = 2^7 \cdot 3$	2	6	407	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.3	2	5	516	0.5	$\{32^6, 144, \infty\}$	$[5, 1/3]$
$x^{12} + t^2x^{11} + t^5x + t$	$864 = 2^5 \cdot 3^3$	6	4	380	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	0.3	6	6	2372	1.2	$\{12^8, 104^3, \infty\}$	$[23/3, 0]$
$x^{12} + (t^3 + t^5)x^5 + t^5x + t$	$3456 = 2^7 \cdot 3^3$	6	5	1748	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	1.4	6	6	3764	2.8	$\{48^8, 512^3, \infty\}$	$[29/3, 0]$
$x^{12} + t^3x^5 + t^5x + t$	$3456 = 2^7 \cdot 3^3$	6	5	1748	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	0.9	6	6	3764	1.8	$\{48^8, 512^3, \infty\}$	$[29/3, 0]$
$x^{12} + (t^3 + t^5)x^5 + t^5x + t$	$3456 = 2^7 \cdot 3^3$	6	5	1748	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	1.3	6	6	3764	2.9	$\{48^8, 512^3, \infty\}$	$[29/3, 0]$
$x^{12} + t^3x^5 + t^5x + t$	$3456 = 2^7 \cdot 3^3$	6	5	1748	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	1.0	6	6	3764	1.9	$\{48^8, 512^3, \infty\}$	$[29/3, 0]$
$x^{12} + t^9x^5 + t$	$3456 = 2^7 \cdot 3^3$	6	5	9328	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	2.4	6	6	38180	102.6	$\{48^8, 1664^3, \infty\}$	$[101/3, 0]$
$x^{16} + t^5x^{14} + t^5x^{13} + t^2x^{12} + tx^8 + t$	$6144 = 2^{11} \cdot 3$	2	6	18644	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	157.4	2	6	22100	344.3	$\{384^8, 1152^4, 3328^3, \infty\}$	$[49/3, 5, 1]$

Table A.2: Eisenstein polynomials with more than one segments over $\mathbb{F}_3((t))$ - Ver1 to Ver2

f	**|Spl(f)|**			**Prec**	**[T:K]**	**T**	**T2**			**Prec**	**T1**		**Slopes**
	$= \lvert\mathbb{F}_{q^{l_2}}((u_{r_2}))\rvert$	l_2	r_2	(u_{r_2})	$= \mathfrak{f}(T/K)e(T/K)$	$(l_T, r_T, t = u^e_{r_T})$	(sec)	l_1	$\mathbf{r}_1$	(u_{r_1})	(sec)	$v_{u_r}(\alpha_i - \alpha)$	
$x^9+(2t^2+2t^{12})x^3+t^6x+2t$	$27 = 3^3$	1	2	669	1	$(1,0,t=t)$	0.2	1	2	1159	0.2	$\{9^6, 54^2, \infty\}$	$[17,2]$
$x^6+tx^5+t^5x+t$	$72 = 2^3\cdot 3^2$	2	4	418	$8 = (2)\cdot(2^2)$	$(2,2,t=2u_2^4)$	0.0	2	4	470	0.2	$\{6^3, 21^2, \infty\}$	$[5/2,0]$
$x^9+(t^2+t^3+t^{30})x^6+(t^2+t^5)x+t$	$72 = 2^3\cdot 3^2$	2	3	367	$8 = (2)\cdot(2^2)$	$(2,2,t=w^2u_2^4)$	0.0	2	3	432	0.4	$\{9^8, \infty\}$	$[5/4]$
$x^9+(t^2+t^3+t^4+t^5)x^8+(t^2+t^3+t^{30})x^6+(t^2+t^5)x+t$	$72 = 2^3\cdot 3^2$	2	3	367	$8 = (2)\cdot(2^2)$	$(2,2,t=w^2u_2^4)$	0.1	2	3	432	0.5	$\{9^8, \infty\}$	$[5/4]$
$x^9+t^3x^8+2t^2x^3+t^6x+t$	$81 = 3^4$	1	3	798	1	$(1,0,t=t)$	0.7	1	3	750	0.5	$\{27^6, 72^2, \infty\}$	$[7,2]$
$x^9+t^6x^8+t^5x^3+t$	$108 = 2^2\cdot 3^3$	2	4	1061	$4 = (2)\cdot(2)$	$(2,2,t=2u_2^2)$	2.8	2	3	1081	4.3	$\{45^6, 48^2, \infty\}$	$[7,13/2]$
$x^9+(t^6+t^8)x^8+(t^3+t^5)x^3+t$	$108 = 2^2\cdot 3^3$	2	4	1311	$4 = (2)\cdot(2)$	$(2,2,t=2u_2^2)$	5.8	2	4	1641	11.2	$\{27^6, 102^2, \infty\}$	$[16,7/2]$
$x^9+t^5x^6+(t^2+t^3+t^{23})x^3+t^5x+t$	$108 = 2^2\cdot 3^3$	2	4	1005	$4 = (2)\cdot(2)$	$(2,2,t=2u_2^2)$	2.1	2	4	998	4.5	$\{18^6, 81^2, \infty\}$	$[25/2,2]$
$x^{12}+t^9x^5+t$	$144 = 2^4\cdot 3^2$	2	4	4505	$16 = (2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	5.2	2	4	9414	64.8	$\{6^9, 309^2, \infty\}$	$[101/2,0]$
$x^{12}+t^3x^5+t^5x+t$	$144 = 2^4\cdot 3^2$	2	4	1481	$16 = (2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	1.8	2	4	1144	1.8	$\{6^9, 93^2, \infty\}$	$[29/2,0]$
$x^9+(t^3+t^4+t^{10})x^8+(t^2+t^3+t^{50})x^2+(t^5+t^{15})x+t$	$144 = 2^4\cdot 3^2$	2	3	503	$16 = (2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	0.2	2	3	328	0.2	$\{19^8, \infty\}$	$[11/8]$
$x^9+t^2x^8+tx^5+tx^3+t$	$162 = 2\cdot 3^4$	3	4	305	$2 = (1)\cdot(2)$	$(1,1,t=2u_1^2)$	0.3	3	3	160	0.2	$\{9^6, 12^2, \infty\}$	$[1,1/2]$
$x^9+t^4x^8+t^3x^5+tx^3+t$	$162 = 2\cdot 3^4$	1	4	646	$2 = (1)\cdot(2)$	$(1,1,t=2u_1^2)$	0.6	1	3	693	0.9	$\{27^6, 198^2, \infty\}$	$[10,1/2]$
$x^9+t^4x^8+tx^3+t$	$324 = 2^2\cdot 3^4$	2	5	830	$4 = (2)\cdot(2)$	$(2,2,t=2u_2^2)$	4.2	2	4	999	6.8	$\{27^6, 306^2, \infty\}$	$[16,1/2]$
$x^9+t^5x^7+t^5x^5+t^2x^3+t$	$324 = 2^2\cdot 3^4$	2	5	1147	$4 = (2)\cdot(2)$	$(2,2,t=u_2^2)$	6.8	2	5	1432	9.0	$\{54^6, 279^2, \infty\}$	$[29/2,2]$
$x^9+t^4x^8+t^5x^5+tx^3+t$	$324 = 2^2\cdot 3^4$	2	5	1011	$4 = (2)\cdot(2)$	$(2,2,t=2u_2^2)$	6.8	2	4	1336	11.2	$\{27^6, 306^2, \infty\}$	$[16,1/2]$
$x^{12}+tx^3+t^3x+t$	$1296 = 2^4\cdot 3^4$	2	6	3260	$16 = (2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	48.8	2	6	5366	224.4	$\{54^9, 729^2, \infty\}$	$[25/2,0]$
$x^{12}+tx^3+t^2x+t$	$1296 = 2^4\cdot 3^4$	2	6	1964	$16 = (2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	11.9	2	6	2882	53.3	$\{54^9, 405^2, \infty\}$	$[13/2,0]$
$x^{12}+t^2x^9+t^5x+t$	$1296 = 2^4\cdot 3^4$	2	6	5852	$16 = (2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	115.5	2	6	10334	457.2	$\{54^9, 1377^2, \infty\}$	$[49/2,0]$
$x^{12}+tx^9+t^5x+t$	$1296 = 2^4\cdot 3^4$	2	6	5852	$16 = (2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	281.4	2	6	10334	1029.1	$\{54^9, 1377^2, \infty\}$	$[49/2,0]$
$x^{15}+t^3x^2+t$	$1620 = 2^2\cdot 3^4\cdot 5$	4	6	2450	$20 = (2^2)\cdot(5)$	$(4,2,t=u_2^5)$	9.0	4	5	3564	18.1	$\{27^{12}, 459^2, \infty\}$	$[16,0]$
$x^{15}+t^4x+t$	$1620 = 2^2\cdot 3^4\cdot 5$	4	6	3395	$20 = (2^2)\cdot(5)$	$(4,2,t=u_2^5)$	10.4	4	5	5022	17.3	$\{27^{12}, 648^2, \infty\}$	$[23,0]$
$x^{15}+t^3x+t$	$3240 = 2^3\cdot 3^4\cdot 5$	4	6	4880	$40 = (2^2)\cdot(2\cdot 5)$	$(4,2,t=u_2^{10})$	8.6	4	6	6804	12.0	$\{54^{12}, 891^2, \infty\}$	$[31/2,0]$
$x^{15}+tx^{14}+t^2x+t$	$4860 = 2^2\cdot 3^5\cdot 5$	4	7	1718	$20 = (2^2)\cdot(5)$	$(4,2,t=u_2^5)$	6.9	4	6	2608	51.9	$\{81^{12}, 648^2, \infty\}$	$[7,0]$
$x^{15}+t^2x^8+tx^3+t$	$9720 = 2^3\cdot 3^5\cdot 5$	4	7	11360	$40 = (2^2)\cdot(2\cdot 5)$	$(4,2,t=u_2^{10})$	1208.4	4	7	7796	1313.2	$\{162^{12}, 2025^2, \infty\}$	$[23/2,0]$
$x^{21}+tx^2+t$	$30618 = 2\cdot 3^7\cdot 7$	6	8	6095	$42 = (2\cdot 3)\cdot(7)$	$(6,2,t=u_2^7)$	2545.4	6	7	17496	8697.2	$\{243^{18}, 486^2, \infty\}$	$[1,0]$
$x^{21}+tx^{15}+t^2x+t$	$30618 = 2\cdot 3^7\cdot 7$	6	8	9983	$42 = (2\cdot 3)\cdot(7)$	$(6,2,t=u_2^7)$	6711.0	6	7	29160	22755.7	$\{243^{18}, 2916^2, \infty\}$	$[11,0]$

A.2 Splitting Fields: Comparison of Version 2 to Version 3

Table A.3: Eisenstein polynomials with more than one segments over $\mathbb{F}_2((t))$ - Ver3 to Ver2

f	**|Spl(f)|** $= \|\mathbb{F}_{q^l}((u_r))\|$	**l**	**r**	**Prec** (u_r)	**[T:K]** $= \mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	l_2	**r**$_2$	**Prec** (u_{r_2})	**T2** (sec)	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^4+(t+t^2)x^2+t^2x+t$	$4=2^2$	1	1	200	1	$(1,0,t=t)$	0.1	1	1	230	0.0	$\{2^2,4,\infty\}$	$[3,1]$
$x^4+t^2x^2+t^6x+t$	$8=2^3$	2	2	220	1	$(1,0,t=t)$	0.0	2	2	220	0.0	$\{4^2,16,\infty\}$	$[15,3]$
$x^{12}+tx^9+t^5x+t$	$24=2^3\cdot 3$	2	3	383	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	2	3	383	0.1	$\{1^8,4^3,\infty\}$	$[3,0]$
$x^8+t^5x^7+t^2x^6+t^2x^5+tx^4+t$	$32=2^5$	2	3	328	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	0.2	2	3	620	0.2	$\{4^4,8^3,\infty\}$	$[3,1]$
$x^6+t^3x^5+t^2x^3+t^2x+t$	$48=2^4\cdot 3$	2	5	397	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.3	2	5	411	0.2	$\{4^4,32,\infty\}$	$[7,0]$
$x^8+t^{11}x^7+t^4x^6+tx^4+t$	$64=2^6$	1	3	1254	1	$(1,0,t=t)$	3.4	1	3	1337	10.3	$\{16^4,112^2,464,\infty\}$	$[57,13,1]$
$x^8+t^7x^7+t^{17}x^6+t^2x^4+t^4x^2+t$	$64=2^6$	1	3	886	1	$(1,0,t=t)$	1.4	1	3	1642	6.1	$\{32^4,64^2,240,\infty\}$	$[29,7,3]$
$x^8+t^9x^7+t^4x^6+t^3x^4+t$	$96=2^5\cdot 3$	3	4	829	$3=(3)\cdot(1)$	$(3,1,t=u_1)$	0.8	3	4	911	1.0	$\{24^6,168,\infty\}$	$[41,5]$
$x^{18}+tx^9+t^2x+t$	$108=2^2\cdot 3^3$	6	3	160	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	0.2	6	3	819	0.5	$\{1^{16},10,\infty\}$	$[9,0]$
$x^8+t^{11}x^7+t^{15}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	1657	1	$(1,0,t=t)$	6.9	1	4	2701	15.3	$\{64^4,128^2,992,\infty\}$	$[61,7,3]$
$x^8+t^{15}x^7+t^{25}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	3423	1	$(1,0,t=t)$	36.2	1	4	4963	85.0	$\{64^4,128^2,1504,\infty\}$	$[93,7,3]$
$x^8+t^{13}x^7+t^{17}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	2223	1	$(1,0,t=t)$	14.8	1	4	3515	30.2	$\{64^4,128^2,1248,\infty\}$	$[77,7,3]$
$x^8+t^{13}x^7+t^{25}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	2855	1	$(1,0,t=t)$	24.6	1	4	4147	47.9	$\{64^4,128^2,1248,\infty\}$	$[77,7,3]$
$x^8+t^9x^7+t^6x^6+t^5x^4+t$	$192=2^6\cdot 3$	2	5	1515	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	2.9	2	5	2279	5.4	$\{104^6,312,\infty\}$	$[25,23/3]$
$x^8+t^9x^7+t^{15}x^6+tx^4+t$	$192=2^6\cdot 3$	2	4	1776	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	1.5	2	4	3045	1.8	$\{24^4,280^3,\infty\}$	$[67/3,1]$
$x^8+t^8x^4+t^{16}x+t$	$192=2^6\cdot 3$	2	4	2014	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	1.5	2	4	3998	2.1	$\{192^4,256^3,\infty\}$	$[61/3,15]$
$x^{18}+t^3x^{11}+t^3x^8+t^2x^3+t^5x+t$	$216=2^3\cdot 3^3$	6	4	1372	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	2.8	6	4	1779	4.9	$\{2^{16},44,\infty\}$	$[21,0]$
$x^{18}+t^3x^{17}+t^3x^9+t^2x^3+t^5x+t$	$216=2^3\cdot 3^3$	6	4	1372	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	3.0	6	4	1779	4.2	$\{2^{16},44,\infty\}$	$[21,0]$
$x^{10}+t^2x^8+tx^3+t$	$320=2^6\cdot 5$	4	6	367	$20=(2^2)\cdot(5)$	$(4,2,t=u_2^5)$	0.3	4	6	517	0.3	$\{8^8,32,\infty\}$	$[3,0]$
$x^{14}+tx+t$	$336=2^4\cdot 3\cdot 7$	6	6	311	$21=(3)\cdot(7)$	$(3,2,t=u_2^7)$	0.4	6	6	547	0.5	$\{4^{12},8,\infty\}$	$[1,0]$
$x^8+t^{13}x^7+t^{25}x^6+t^8x^4+t^4x^2+t$	$384=2^7\cdot 3$	2	6	6224	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	73.8	2	6	8911	145.0	$\{128^6,1872,\infty\}$	$[77,13/3]$
$x^8+t^{13}x^7+t^{25}x^6+t^{12}x^4+t^8x^2+t$	$384=2^7\cdot 3$	2	6	4013	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	25.8	2	6	6639	68.4	$\{256^6,1104,\infty\}$	$[45,29/3]$
$x^8+t^{19}x^7+t^{25}x^6+t^{12}x^4+t^8x^2+t$	$384=2^7\cdot 3$	2	6	7457	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	97.7	2	6	11391	225.8	$\{256^6,2256,\infty\}$	$[93,29/3]$
$x^8+t^{19}x^7+t^{25}x^6+t^{10}x^4+t^8x^2+t$	$384=2^7\cdot 3$	2	6	7457	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	113.0	2	6	11391	241.4	$\{256^6,2256,\infty\}$	$[93,29/3]$
$x^{16}+tx^{15}+tx^6+tx^4+t$	$1536=2^9\cdot 3$	2	7	548	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	1.1	2	7	946	2.0	$\{64^{12},96^2,480,\infty\}$	$[9,1,1/3]$
$x^{16}+tx^{15}+tx^4+t$	$1536=2^9\cdot 3$	2	5	531	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	2.3	2	5	5184	4.5	$\{64^{12},224^3,\infty\}$	$[11/3,1/3]$
$x^{16}+t^5x^{14}+t^7x^{13}+t^2x^{12}+tx^8+t$	$2048=2^{11}$	2	6	2375	1	$(1,0,t=t)$	61.1	2	6	23636	724.7	$\{128^8,384^4,1664^2,2048,\infty\}$	$[31,25,5,1]$
$x^{18}+tx+t$	$3456=2^7\cdot 3^3$	6	8	884	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	1.2	6	8	1265	3.1	$\{32^{16},64,\infty\}$	$[1,0]$
$x^{16}+t^3x^{14}+t^5x^{13}+t^2x^{12}+tx^8+t$	$4096=2^{12}$	2	8	4244	1	$(1,0,t=t)$	29.1	2	8	14568	571.7	$\{256^8,768^4,1280^2,4096,\infty\}$	$[31,9,5,1]$
$x^{16}+t^5x^{14}+t^5x^{13}+t^2x^{12}+tx^8+t$	$6144=2^{11}\cdot 3$	2	6	3668	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	7.8	2	6	18644	156.6	$\{384^8,1152^4,3328^3,\infty\}$	$[49/3,5,1]$
$x^{18}+tx^{17}+tx^9+t^2x^3+t^3x+t$	$6912=2^8\cdot 3^3$	6	9	6356	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	54.5	6	9	6356	51.1	$\{64^{16},640,\infty\}$	$[9,0]$
$x^{16}+tx^9+tx^7+tx^6+tx^5+tx^4+tx^3+tx^2+t$	$43008=2^{11}\cdot 3\cdot 7$	6	10	1364	$21=(3)\cdot(7)$	$(3,2,t=u_2^7)$	2.7	6	10	1926	5.9	$\{512^{14},896,\infty\}$	$[1,1/7]$

Table A.4: Eisenstein polynomials with more than one segments over $\mathbb{F}_3((t))$ - Ver3 to Ver2

f	**|Spl(f)|** $= \|\mathbb{F}_{q^l}((u_r))\|$	**l**	**r**	**Prec** (u_r)	**[T:K]** $= \mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	l_2	$\mathbf{r}_2$	**Prec** (u_{r_2})	**T2** (sec)	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^9 + 2t^2x^3 + t^{10}x + 2t$	$27 = 3^3$	1	2	786	1	$(1, 0, t = t)$	0.3	1	2	884	0.3	$\{9^6, 108^2, \infty\}$	$[35, 2]$
$x^9 + t^2x^3 + t^4x + t$	$54 = 2 \cdot 3^3$	2	3	155	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.2	2	3	479	0.2	$\{9^6, 27^2, \infty\}$	$[8, 2]$
$x^6 + tx + t$	$72 = 2^3 \cdot 3^2$	2	4	252	$8 = (2) \cdot (2^2)$	$(2, 2, t = 2u_2^4)$	0.2	2	4	334	0.2	$\{6^3, 9^2, \infty\}$	$[1/2, 0]$
$x^9 + t^3x^8 + 2t^2x^3 + t^6x + t$	$81 = 3^4$	1	3	428	1	$(1, 0, t = t)$	0.3	1	3	798	0.7	$\{27^6, 72^2, \infty\}$	$[7, 2]$
$x^9 + t^6x^8 + t^5x^3 + t$	$108 = 2^2 \cdot 3^3$	2	4	600	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	1.4	2	4	1061	2.7	$\{45^6, 48^2, \infty\}$	$[7, 13/2]$
$x^9 + (t^6 + t^8)x^8 + (t^3 + t^5)x^3 + t$	$108 = 2^2 \cdot 3^3$	2	4	995	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	4.3	2	4	1311	5.6	$\{27^6, 102^2, \infty\}$	$[16, 7/2]$
$x^{12} + t^2x^{11} + t^5x + t$	$144 = 2^4 \cdot 3^2$	2	4	1118	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	1.3	2	4	1298	1.6	$\{6^9, 75^2, \infty\}$	$[23/2, 0]$
$x^{12} + (t^3 + t^5)x^5 + t^5x + t$	$144 = 2^4 \cdot 3^2$	2	4	1277	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	4.0	2	4	1481	5.1	$\{6^9, 93^2, \infty\}$	$[29/2, 0]$
$x^9 + t^2x^8 + tx^5 + tx^3 + t$	$162 = 2 \cdot 3^4$	3	4	318	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	0.3	3	4	305	0.2	$\{9^6, 12^2, \infty\}$	$[1, 1/2]$
$x^9 + t^4x^8 + tx^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	777	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	4.5	2	5	830	4.2	$\{27^6, 306^2, \infty\}$	$[16, 1/2]$
$x^9 + t^5x^7 + t^5x^5 + t^2x^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	870	$4 = (2) \cdot (2)$	$(2, 2, t = u_2^2)$	5.8	2	5	1147	6.3	$\{54^6, 279^2, \infty\}$	$[29/2, 2]$
$x^{12} + tx^3 + t^5x + t$	$1296 = 2^4 \cdot 3^4$	2	6	5852	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	236.0	2	6	5852	316.8	$\{54^9, 1377^2, \infty\}$	$[49/2, 0]$
$x^{12} + t^2x^9 + t^5x + t$	$1296 = 2^4 \cdot 3^4$	2	6	5852	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	77.4	2	6	5852	113.2	$\{54^9, 1377^2, \infty\}$	$[49/2, 0]$
$x^{12} + tx^9 + t^5x + t$	$1296 = 2^4 \cdot 3^4$	2	6	5852	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	238.2	2	6	5852	433.1	$\{54^9, 1377^2, \infty\}$	$[49/2, 0]$
$x^{27} + t^7x^9 + t^{13}x^3 + t^{15}x + t$	$1458 = 2 \cdot 3^6$	3	4	3303	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	1166.1	3	4	31104	2765.7	$\{189^{18}, 486^8, \infty\}$	$[26, 19/2]$
$x^{15} + t^2x + t$	$1620 = 2^2 \cdot 3^4 \cdot 5$	4	6	1370	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	0.4	4	6	1370	0.4	$\{27^{12}, 243^2, \infty\}$	$[8, 0]$
$x^{27} + t^3x^9 + t^2x^3 + t^4x + t$	$1944 = 2^3 \cdot 3^5$	2	5	2246	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^6u_2^4)$	62.8	2	5	3888	164.4	$\{81^{24}, 972^2, \infty\}$	$[26, 5/4]$
$x^{27} + t^2x^3 + t^4x + t$	$1944 = 2^3 \cdot 3^5$	2	5	2246	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^6u_2^4)$	41.8	2	5	3888	94.7	$\{81^{24}, 972^2, \infty\}$	$[26, 5/4]$
$x^{27} + t^7x^9 + t^{10}x^3 + t^{11}x + t$	$2916 = 2^2 \cdot 3^6$	6	4	1945	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	1369.4	6	4	15552	2100.7	$\{189^{18}, 243^8, \infty\}$	$[25/2, 19/2]$
$x^{15} + t^4x^3 + tx + t$	$3240 = 2^3 \cdot 3^4 \cdot 5$	4	6	830	$40 = (2^2) \cdot (2 \cdot 5)$	$(4, 2, t = u_2^{10})$	6.9	4	6	1128	13.5	$\{54^{12}, 81^2, \infty\}$	$[1/2, 0]$
$x^{15} + tx + t$	$3240 = 2^3 \cdot 3^4 \cdot 5$	4	6	830	$40 = (2^2) \cdot (2 \cdot 5)$	$(4, 2, t = u_2^{10})$	4.8	4	6	830	7.7	$\{54^{12}, 81^2, \infty\}$	$[1/2, 0]$
$x^{15} + t^2x^5 + tx + t$	$3240 = 2^3 \cdot 3^4 \cdot 5$	4	6	830	$40 = (2^2) \cdot (2 \cdot 5)$	$(4, 2, t = u_2^{10})$	5.0	4	6	830	8.0	$\{54^{12}, 81^2, \infty\}$	$[1/2, 0]$
$x^{15} + tx^3 + tx + t$	$3240 = 2^3 \cdot 3^4 \cdot 5$	4	6	830	$40 = (2^2) \cdot (2 \cdot 5)$	$(4, 2, t = u_2^{10})$	4.8	4	6	830	7.9	$\{54^{12}, 81^2, \infty\}$	$[1/2, 0]$
$x^{24} + tx + t$	$5184 = 2^6 \cdot 3^4$	4	6	1316	$64 = (2^2) \cdot (2^4)$	$(4, 2, t = 2u_2^{16})$	9.9	4	6	1798	55.6	$\{54^{21}, 81^2, \infty\}$	$[1/2, 0]$
$x^{24} + t^2x^7 + tx + t$	$5184 = 2^6 \cdot 3^4$	4	6	1316	$64 = (2^2) \cdot (2^4)$	$(4, 2, t = 2u_2^{16})$	10.3	4	6	1316	21.5	$\{54^{21}, 81^2, \infty\}$	$[1/2, 0]$
$x^{27} + 2tx^{13} + 2tx^{12} + tx^{10} + 2tx^7 + 2tx^6 + 2t$	$5832 = 2^3 \cdot 3^6$	6	6	461	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^6u_2^4)$	12.0	6	6	953	15.0	$\{45^{24}, 54^2, \infty\}$	$[1/2, 1/4]$
$x^{27} + 2tx^{20} + tx^{18} + tx^{14} + tx^{12} + 2tx^8 + 2tx^6 + 2t$	$5832 = 2^3 \cdot 3^6$	6	6	466	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	9.4	6	6	961	13.0	$\{45^{24}, 72^2, \infty\}$	$[1, 1/4]$
$x^{21} + tx^2 + t$	$30618 = 2 \cdot 3^7 \cdot 7$	6	8	5123	$42 = (2 \cdot 3) \cdot (7)$	$(6, 2, t = u_2^7)$	867.4	6	8	7553	3108.9	$\{243^{18}, 486^2, \infty\}$	$[1, 0]$
$x^{27} + tx^{26} + tx^7 + tx^3 + t$	$104976 = 2^4 \cdot 3^8$	2	8	7796	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	824.1	2	8	34992	3730.2	$\{2187^{24}, 5832^2, \infty\}$	$[2, 1/8]$
$x^{27} + tx^{20} + tx^7 + tx^3 + t$	$104976 = 2^4 \cdot 3^8$	2	8	7796	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	669.8	2	8	34992	3663.1	$\{2187^{24}, 5832^2, \infty\}$	$[2, 1/8]$

A.3 Splitting Fields and Galois Groups

Table A.5: Eisenstein polynomials with more than one segments over $\mathbb{F}_2((t))$

	|Spl(f)|			**Prec**	**[T:K]**	**T**	**T3**		**T4**			
f	$= \lvert\mathbb{F}_{q^l}((u_r))\rvert$	**l**	**r**	(u_r)	$=\mathfrak{f}(T/K)e(T/K)$	$(l_T, r_T, t = u_{r_T}^e)$	(sec)	**Gal(f)**	(sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^4 + (t+t^2)x^2 + t^2x + t$	$4 = 2^2$	1	1	200	1	$(1, 0, t = t)$	0.0	C_4	0.0	2	$\{2^2, 4, \infty\}$	$[3, 1]$
$x^8 + (t^6+t^{16}+t^{18}+t^{28}+t^{32})x^6 + (t^7+t^{11}+t^{12}+t^{16}+t^{17}+t^{22}+t^{25}+t^{27}+t^{29}+t^{30}+t^{32}+t^{34})x^5 + (t^2+t^3+t^5+t^6+t^9+t^{10}+t^{11}+t^{15}+t^{16}+t^{17}+t^{19}+t^{20}+t^{21}+t^{25}+t^{29}+t^{35}+t^{36}+t^{42}+t^{43}+t^{45}+t^{47}+t^{50}+t^{52}+t^{56}+t^{60}+t^{64})x^4 + (t^8+t^{12}+t^{18}+t^{20}+t^{22}+t^{23}+t^{25}+t^{26}+t^{28}+t^{30}+t^{32}+t^{34}+t^{35}+t^{37}+t^{38}+t^{40}+t^{44}+t^{46}+t^{47}+t^{48}+t^{49}+t^{52}+t^{55}+t^{57}+t^{58}+t^{60}+t^{62}+t^{64}+t^{66}+t^{70})x^3 + (t^4+t^6+t^7+t^8+t^9+t^{10}+t^{11}+t^{13}+t^{14}+t^{16}+t^{23}+t^{24}+t^{26}+t^{27}+t^{28}+t^{29}+t^{30}+t^{36}+t^{37}+t^{38}+t^{39}+t^{41}+t^{46}+t^{47}+t^{48}+t^{49}+t^{50}+t^{54}+t^{55}+t^{56}+t^{57}+t^{67}+t^{68}+t^{70}+t^{73}+t^{76}+t^{77}+t^{78}+t^{79}+t^{80}+t^{82}+t^{84}+t^{88}+t^{96})x^2 + (t^9+t^{12}+t^{13}+t^{15}+t^{18}+t^{21}+t^{23}+t^{24}+t^{27}+t^{30}+t^{31}+t^{32}+t^{33}+t^{35}+t^{38}+t^{41}+t^{43}+t^{44}+t^{47}+t^{49}+t^{58}+t^{62}+t^{64}+t^{66}+t^{68}+t^{70}+t^{71}+t^{74}+t^{78}+t^{80}+t^{82}+t^{83}+t^{89}+t^{90}+t^{91}+t^{92}+t^{93}+t^{94})x + t + t^4+t^{10}+t^{12}+t^{13}+t^{14}+t^{15}+t^{16}+t^{17}+t^{19}+t^{20}+t^{24}+t^{26}+t^{27}+t^{30}+t^{31}+t^{35}+t^{37}+t^{39}+t^{41}+t^{43}+t^{45}+t^{46}+t^{47}+t^{48}+t^{50}+t^{52}+t^{53}+t^{55}+t^{56}+t^{59}+t^{60}+t^{61}+t^{62}+t^{64}+t^{66}+t^{68}+t^{69}+t^{70}+t^{72}+t^{73}+t^{74}+t^{75}+t^{76}+t^{79}+t^{80}+t^{81}+t^{82}+t^{83}+t^{86}+t^{90}+t^{91}+t^{94}+t^{95}+t^{96}$	$8 = 2^3$	1	1	543	1	$(1, 0, t = t)$	0.2	Q_8	0.2	4	$\{4^4, 8^2, 28, \infty\}$	$[27, 7, 3]$
$x^4 + t^4x^2 + t^8x + t$	$8 = 2^3$	1	2	267	1	$(1, 0, t = t)$	0.2	D_4	0.1	3	$\{16^2, 32, \infty\}$	$[15, 7]$
$x^4 + tx^2 + t^2x + t$	$8 = 2^3$	2	2	200	1	$(1, 0, t = t)$	0.1	D_4	0.0	3	$\{2^2, 4, \infty\}$	$[3, 1]$
$x^4 + t^4x^2 + t^{10}x + t$	$8 = 2^3$	1	2	413	1	$(1, 0, t = t)$	0.1	D_4	0.2	3	$\{16^2, 48, \infty\}$	$[23, 7]$
$x^4 + t^4x^2 + t^{14}x + t$	$8 = 2^3$	1	2	801	1	$(1, 0, t = t)$	0.3	D_4	0.7	3	$\{16^2, 80, \infty\}$	$[39, 7]$
$x^4 + t^8x^3 + t^4x^2 + t^{14}x + t$	$8 = 2^3$	1	2	407	1	$(1, 0, t = t)$	0.2	D_4	0.1	3	$\{16^2, 36, \infty\}$	$[17, 7]$

Continued on next page

Table A.5 – continued from previous page

	|Spl(f)|			**Prec**	**[T:K]**		**T3**		**T4**			
f	$= \lvert\mathbb{F}_{q^l}((u_r))\rvert$	**l**	**r**	(u_r)	$=\mathfrak{f}(T/K)e(T/K)$	**T**	(seq)	**Gal(f)**	(seq)	$\#\sigma$	$v_{u_r}(\alpha_i-\alpha)$	**Slopes**
$x^{12}+tx^3+t^5x+t$	$24=2^3\cdot 3$	2	3	323	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	S_4	0.2	4	$\{1^8,2^3,\infty\}$	$[1,0]$
$x^{12}+tx^3+t^3x+t$	$24=2^3\cdot 3$	2	3	293	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	S_4	0.1	4	$\{1^8,2^3,\infty\}$	$[1,0]$
$x^{12}+tx^3+t^2x+t$	$24=2^3\cdot 3$	2	3	293	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	S_4	0.1	4	$\{1^8,2^3,\infty\}$	$[1,0]$
x^6+tx^5+tx+t	$24=2^3\cdot 3$	2	4	236	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	S_4	0.1	3	$\{2^4,4,\infty\}$	$[1,0]$
x^6+tx+t	$24=2^3\cdot 3$	2	4	236	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	S_4	0.0	3	$\{2^4,4,\infty\}$	$[1,0]$
$x^{12}+t^2x^9+t^5x+t$	$24=2^3\cdot 3$	2	3	503	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	S_4	0.2	4	$\{1^8,8^3,\infty\}$	$[7,0]$
$x^{12}+tx^9+t^5x+t$	$24=2^3\cdot 3$	2	3	383	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.2	$12T13$	0.2	4	$\{1^8,4^3,\infty\}$	$[3,0]$
$x^8+t^3x^7+t^2x^6+tx^4+t$	$32=2^5$	2	3	152	1	$(1,0,t=t)$	0.2	$8T20$	0.1	5	$\{4^4,12^2,20,\infty\}$	$[9,5,1]$
$x^8+t^3x^7+t^2x^6+t^2x^5+tx^4+t$	$32=2^5$	2	3	213	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	0.2	$8T20$	0.1	5	$\{4^4,8^3,\infty\}$	$[3,1]$
$x^8+t^5x^7+t^2x^6+t^2x^5+tx^4+t$	$32=2^5$	2	3	328	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	0.2	$8T20$	0.2	5	$\{4^4,8^3,\infty\}$	$[3,1]$
$x^6+tx^5+t^3x^3+t^2x+t$	$48=2^4\cdot 3$	2	5	294	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.3	$6T11$	0.2	4	$\{4^4,24,\infty\}$	$[5,0]$
$x^8+t^5x^7+t^2x^6+tx^4+t$	$64=2^6$	2	4	443	1	$(1,0,t=t)$	0.7	$8T29$	0.8	4	$\{8^4,24^2,104,\infty\}$	$[25,5,1]$
$x^8+t^7x^7+t^2x^6+tx^4+t$	$64=2^6$	2	4	753	1	$(1,0,t=t)$	1.6	$8T29$	1.8	4	$\{8^4,24^2,168,\infty\}$	$[41,5,1]$
$x^8+t^9x^7+t^2x^6+tx^4+t$	$64=2^6$	2	4	1127	1	$(1,0,t=t)$	3.7	$8T29$	4.5	4	$\{8^4,24^2,232,\infty\}$	$[57,5,1]$
$x^8+t^9x^7+t^4x^6+tx^4+t$	$64=2^6$	1	3	834	1	$(1,0,t=t)$	1.4	$8T29$	32.4	3	$\{16^4,112^2,336,\infty\}$	$[41,13,1]$
$x^8+t^{11}x^7+t^4x^6+tx^4+t$	$64=2^6$	1	3	1254	1	$(1,0,t=t)$	3.5	$8T29$	54.1	3	$\{16^4,112^2,464,\infty\}$	$[57,13,1]$
$x^8+t^{11}x^7+t^6x^6+tx^4+t$	$64=2^6$	2	3	894	1	$(1,0,t=t)$	1.3	$8T29$	1.7	3	$\{8^4,88^2,168,\infty\}$	$[41,21,1]$
$x^8+t^{13}x^7+t^8x^6+tx^4+t$	$64=2^6$	1	3	1004	1	$(1,0,t=t)$	2.0	$8T29$	8.7	5	$\{16^4,240^2,336,\infty\}$	$[41,29,1]$
$x^8+t^7x^7+t^{17}x^6+t^2x^4+t^4x^2+t$	$64=2^6$	1	3	886	1	$(1,0,t=t)$	1.5	$8T29$	2.5	3	$\{32^4,64^2,240,\infty\}$	$[29,7,3]$
$x^8+t^9x^7+t^4x^6+tx^4+t$	$64=2^6$	1	3	834	1	$(1,0,t=t)$	1.5	$8T29$	34.3	3	$\{16^4,112^2,336,\infty\}$	$[41,13,1]$
$x^8+t^5x^7+t^2x^6+tx^4+t$	$64=2^6$	2	4	443	1	$(1,0,t=t)$	0.7	$8T29$	0.9	4	$\{8^4,24^2,104,\infty\}$	$[25,5,1]$
$x^8+t^2x^6+tx^4+t^4x^2+t^4x+t$	$64=2^6$	2	4	187	1	$(1,0,t=t)$	0.3	$8T26$	0.2	4	$\{8^4,24^2,48,\infty\}$	$[11,5,1]$
$x^{18}+tx^9+t^2x+t$	$108=2^2\cdot 3^3$	6	3	160	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	0.2	$18T45$	0.2	4	$\{1^{16},10,\infty\}$	$[9,0]$
$x^{18}+tx^9+t^3x^3+t^3x+t$	$108=2^2\cdot 3^3$	6	3	160	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	0.3	$18T45$	0.3	4	$\{1^{16},10,\infty\}$	$[9,0]$
$x^{18}+tx^9+t^2x^3+t^3x+t$	$108=2^2\cdot 3^3$	6	3	160	$54=(2\cdot 3)\cdot(3^2)$	$(6,2,t=u_2^9)$	0.4	$18T45$	0.3	4	$\{1^{16},10,\infty\}$	$[9,0]$
$x^8+t^{11}x^7+t^{15}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	1657	1	$(1,0,t=t)$	7.9	$8T35$	77.4	3	$\{64^4,128^2,992,\infty\}$	$[61,7,3]$
$x^8+t^{15}x^7+t^{25}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	3423	1	$(1,0,t=t)$	36.2	$8T35$	193.4	3	$\{64^4,128^2,1504,\infty\}$	$[93,7,3]$
$x^8+t^9x^7+t^{17}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	1340	1	$(1,0,t=t)$	4.8	$8T35$	98.3	3	$\{64^4,128^2,736,\infty\}$	$[45,7,3]$
$x^8+t^{13}x^7+t^{17}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	2223	1	$(1,0,t=t)$	15.0	$8T35$	194.9	3	$\{64^4,128^2,1248,\infty\}$	$[77,7,3]$
$x^8+t^{13}x^7+t^{19}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	2381	1	$(1,0,t=t)$	17.2	$8T35$	201.6	3	$\{64^4,128^2,1248,\infty\}$	$[77,7,3]$
$x^8+t^{13}x^7+t^{25}x^6+t^2x^4+t^4x^2+t$	$128=2^7$	1	4	2855	1	$(1,0,t=t)$	24.9	$8T35$	209.3	3	$\{64^4,128^2,1248,\infty\}$	$[77,7,3]$
$x^8+tx^6+(t+t^4)x^3+t^3x^2+t^2x+t$	$168=2^3\cdot 3\cdot 7$	3	3	353	$21=(3)\cdot(7)$	$(3,2,t=u_2^7)$	0.2	$8T36$	0.0	5	$\{10^7,\infty\}$	$[3/7]$
$x^8+t^2x^7+tx^4+t$	$192=2^6\cdot 3$	2	4	420	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.3	$8T41$	0.4	5	$\{24^4,56^3,\infty\}$	$[11/3,1]$

Continued on next page

Table A.5 – continued from previous page

f	**|Spl(f)|**	**l**	**r**	**Prec**	**[T:K]**	**T**	**T3**	**Gal(f)**	**T4**	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
	$= \|\mathbb{F}_{q^l}((u_r))\|$			(u_r)	$=\mathfrak{f}(T/K)e(T/K)$		(seq)		(seq)			
$x^8 + t^9x^7 + t^6x^6 + t^5x^4 + t$	$192 = 2^6 \cdot 3$	2	5	1515	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	2.8	$8T39$	15.5	4	$\{104^6, 312, \infty\}$	$[25, 23/3]$
$x^8 + t^9x^7 + t^{15}x^6 + tx^4 + t$	$192 = 2^6 \cdot 3$	2	4	1776	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	1.5	$8T41$	69.2	5	$\{24^4, 280^3, \infty\}$	$[67/3, 1]$
$x^8 + t^2x^6 + tx^4 + t^4x^2 + t^2x + t$	$192 = 2^6 \cdot 3$	2	4	366	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.3	$8T41$	0.3	5	$\{24^4, 32^3, \infty\}$	$[5/3, 1]$
$x^8 + t^4x^2 + t^8x + t$	$192 = 2^6 \cdot 3$	2	5	1573	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	2.6	$8T39$	534.5	4	$\{64^6, 384, \infty\}$	$[31, 13/3]$
$x^8 + t^8x^4 + t^{16}x + t$	$192 = 2^6 \cdot 3$	2	4	2014	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	1.6	$8T41$	62.9	5	$\{192^4, 256^3, \infty\}$	$[61/3, 15]$
$x^8 + t^2x^6 + tx^4 + tx^2 + t^2x + t$	$192 = 2^6 \cdot 3$	2	5	469	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.5	$8T40$	0.7	4	$\{16^6, 96, \infty\}$	$[7, 1/3]$
$x^8 + t^4x^4 + tx^2 + t^2x + t$	$192 = 2^6 \cdot 3$	2	5	469	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.4	$8T39$	0.5	4	$\{16^6, 96, \infty\}$	$[7, 1/3]$
$x^{18} + t^3x^{17} + t^3x^9 + t^2x^3 + t^5x + t$	$216 = 2^3 \cdot 3^3$	6	4	1372	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	2.9	$18T98$	3.2	4	$\{2^{16}, 44, \infty\}$	$[21, 0]$
$x^{18} + t^3x^{11} + t^3x^8 + t^2x^3 + t^5x + t$	$216 = 2^3 \cdot 3^3$	6	4	1372	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	3.1	$18T98$	3.2	4	$\{2^{16}, 44, \infty\}$	$[21, 0]$
$x^{10} + tx + t$	$320 = 2^6 \cdot 5$	4	6	336	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	0.4	$10T24$	0.3	4	$\{8^8, 16, \infty\}$	$[1, 0]$
$x^8 + t^9x^7 + t^{15}x^6 + tx^2 + t$	$384 = 2^7 \cdot 3$	2	6	4200	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	34.2	$8T44$	1241.1	4	$\{32^6, 1680, \infty\}$	$[69, 1/3]$
$x^8 + t^{13}x^7 + t^{25}x^6 + t^8x^4 + t^4x^2 + t$	$384 = 2^7 \cdot 3$	2	6	6224	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	74.0	$8T44$	2392.5	4	$\{128^6, 1872, \infty\}$	$[77, 13/3]$
$x^8 + t^{13}x^7 + t^{25}x^6 + t^{12}x^4 + t^8x^2 + t$	$384 = 2^7 \cdot 3$	2	6	4013	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	26.6	$8T44$	178.6	4	$\{256^6, 1104, \infty\}$	$[45, 29/3]$
$x^8 + t^{19}x^7 + t^{25}x^6 + t^{12}x^4 + t^8x^2 + t$	$384 = 2^7 \cdot 3$	2	6	7457	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	97.9	$8T44$	1602.0	4	$\{256^6, 2256, \infty\}$	$[93, 29/3]$
$x^8 + t^5x^7 + t^{11}x^6 + t^3x^4 + t^2x^2 + t$	$384 = 2^7 \cdot 3$	2	6	1784	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	5.4	$8T44$	66.9	4	$\{64^6, 720, \infty\}$	$[29, 5/3]$
$x^8 + t^3x^7 + t^2x^6 + tx^4 + tx^2 + t$	$384 = 2^7 \cdot 3$	2	6	572	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.6	$8T44$	22.9	4	$\{32^6, 528, \infty\}$	$[21, 1/3]$
$x^8 + t^5x^7 + t^2x^6 + tx^4 + tx^2 + t$	$384 = 2^7 \cdot 3$	2	6	1638	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	4.8	$8T44$	63.1	4	$\{32^6, 912, \infty\}$	$[37, 1/3]$
$x^8 + tx^7 + tx^2 + t$	$384 = 2^7 \cdot 3$	2	6	468	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.5	$8T44$	1.0	4	$\{32^6, 144, \infty\}$	$[5, 1/3]$
$x^{16} + tx^{15} + tx^6 + tx^4 + t$	$1536 = 2^9 \cdot 3$	2	7	548	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	1.1	$16T1311$	5.8	4	$\{64^{12}, 96^2, 480, \infty\}$	$[9, 1, 1/3]$
$x^{16} + tx^{15} + tx^4 + t$	$1536 = 2^9 \cdot 3$	2	5	531	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	2.2	$16T1317$	4.2	5	$\{64^{12}, 224^3, \infty\}$	$[11/3, 1/3]$
$x^{16} + t^5x^{14} + t^7x^{13} + t^2x^{12} + tx^8 + t$	$2048 = 2^{11}$	2	6	2375	1	$(1, 0, t = t)$	61.7	$16T1467$	1356.2	4	$\{128^8, 384^4, 1664^2, 2048, \infty\}$	$[31, 25, 5, 1]$
$x^{16} + t^7x^{14} + t^9x^{13} + t^4x^{12} + tx^8 + t$	$2048 = 2^{11}$	2	5	2729	1	$(1, 0, t = t)$	43.4	$16T1344$	2660.9	5	$\{128^8, 896^4, 1664^2, 2048, \infty\}$	$[31, 25, 13, 1]$
$x^{16} + t^7x^{15} + t^9x^{10} + t^{11}x^9 + t^3x^8 + t^6x^4 + t$	$2688 = 2^7 \cdot 3 \cdot 7$	3	4	3059	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	7.9	$16T1501$	10.7	6	$\{336^8, 624^7, \infty\}$	$[71/7, 5]$
$x^{12} + t^5x^5 + t$	$3456 = 2^7 \cdot 3^3$	6	5	2900	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	3.1	$12T254$	5.1	5	$\{48^8, 896^3, \infty\}$	$[53/3, 0]$
$x^{12} + t^9x^5 + t$	$3456 = 2^7 \cdot 3^3$	6	5	5499	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	5.8	$12T254$	13.1	5	$\{48^8, 1664^3, \infty\}$	$[101/3, 0]$
$x^{18} + tx + t$	$3456 = 2^7 \cdot 3^3$	6	8	884	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	1.2	$18T433$	1.5	4	$\{32^{16}, 64, \infty\}$	$[1, 0]$
$x^{12} + tx + t$	$3456 = 2^7 \cdot 3^3$	6	5	452	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	1.0	$12T254$	1.2	5	$\{48^8, 64^3, \infty\}$	$[1/3, 0]$
$x^{12} + (t^3 + t^5)x^5 + t^5x + t$	$3456 = 2^7 \cdot 3^3$	6	5	1748	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	4.2	$12T254$	4.6	5	$\{48^8, 512^3, \infty\}$	$[29/3, 0]$
$x^{12} + t^3x^5 + t^5x + t$	$3456 = 2^7 \cdot 3^3$	6	5	1748	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	3.8	$12T254$	4.2	5	$\{48^8, 512^3, \infty\}$	$[29/3, 0]$
$x^{16} + t^5x^{14} + t^5x^{13} + t^2x^{12} + tx^8 + t$	$6144 = 2^{11} \cdot 3$	2	6	3668	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	7.9	$16T1673$	698.7	5	$\{384^8, 1152^4, 3328^3, \infty\}$	$[49/3, 5, 1]$
$x^{24} + tx + t$	$8064 = 2^7 \cdot 3^2 \cdot 7$	6	4	524	$126 = (2 \cdot 3) \cdot (3 \cdot 7)$	$(6, 2, t = u_2^{21})$	18.3	$24T9738$	18.5	5	$\{56^{16}, 64^7, \infty\}$	$[1/7, 0]$

Table A.6: Eisenstein polynomials with more than one segments over $\mathbb{F}_3((t))$

f	**|Spl(f)|**	**l**	**r**	**Prec**	**[T:K]**	**T**	**T3**	**Gal(f)**	**T4**			**Slopes**
	$= \lvert\mathbb{F}_{q^l}((u_r))\rvert$			(u_r)	$= \mathfrak{f}(T/K)e(T/K)$	$(l_T, r_T, t = u_{r_T}^e)$	(sec)		(sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	
$x^9 + (2t^2 + 2t^{12})x^3 + t^6x + 2t$	$27 = 3^3$	1	2	356	1	$(1, 0, t = t)$	0.2	He_3	0.2	3	$\{9^6, 54^2, \infty\}$	$[17, 2]$
$x^9 + 2t^2x^3 + t^{10}x + 2t$	$27 = 3^3$	1	2	786	1	$(1, 0, t = t)$	0.4	He_3	0.4	3	$\{9^6, 108^2, \infty\}$	$[35, 2]$
$x^9 + tx^3 + t^2x + t$	$54 = 2 \cdot 3^3$	1	3	371	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	0.5	$9T11$	0.5	4	$\{9^6, 27^2, \infty\}$	$[7/2, 1/2]$
$x^9 + t^2x^3 + t^4x + t$	$54 = 2 \cdot 3^3$	2	3	155	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.2	$9T11$	0.0	4	$\{9^6, 27^2, \infty\}$	$[8, 2]$
$x^9 + t^4x^3 + t^6x + t$	$54 = 2 \cdot 3^3$	2	3	215	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.2	$9T11$	0.1	4	$\{18^6, 27^2, \infty\}$	$[8, 5]$
$x^6 + tx + t$	$72 = 2^3 \cdot 3^2$	2	4	252	$8 = (2) \cdot (2^2)$	$(2, 2, t = 2u_2^4)$	0.2	$6T13$	0.2	4	$\{6^3, 9^2, \infty\}$	$[1/2, 0]$
$x^6 + t^3x + 2t$	$72 = 2^3 \cdot 3^2$	2	4	486	$8 = (2) \cdot (2^2)$	$(2, 2, t = u_2^4)$	0.2	$6T13$	0.1	4	$\{6^3, 45^2, \infty\}$	$[13/2, 0]$
$x^6 + tx^5 + tx + t$	$72 = 2^3 \cdot 3^2$	2	4	252	$8 = (2) \cdot (2^2)$	$(2, 2, t = 2u_2^4)$	0.3	$6T13$	0.2	4	$\{6^3, 9^2, \infty\}$	$[1/2, 0]$
$x^6 + t^2x^3 + t^2x + t$	$72 = 2^3 \cdot 3^2$	2	4	393	$8 = (2) \cdot (2^2)$	$(2, 2, t = 2u_2^4)$	0.3	$6T13$	0.3	4	$\{6^3, 27^2, \infty\}$	$[7/2, 0]$
$x^6 + t^3x^5 + t^2x^3 + t^2x + t$	$72 = 2^3 \cdot 3^2$	2	4	393	$8 = (2) \cdot (2^2)$	$(2, 2, t = 2u_2^4)$	0.4	$6T13$	0.3	4	$\{6^3, 27^2, \infty\}$	$[7/2, 0]$
$x^6 + tx^5 + t^3x^3 + t^2x + t$	$72 = 2^3 \cdot 3^2$	2	4	369	$8 = (2) \cdot (2^2)$	$(2, 2, t = 2u_2^4)$	0.3	$6T13$	0.3	4	$\{6^3, 21^2, \infty\}$	$[5/2, 0]$
$x^9 + t^3x^8 + 2t^2x^3 + t^6x + t$	$81 = 3^4$	1	3	428	1	$(1, 0, t = t)$	0.4	$9T17$	0.4	3	$\{27^6, 72^2, \infty\}$	$[7, 2]$
$x^9 + t^4x^8 + t^3x^3 + t$	$108 = 2^2 \cdot 3^3$	2	4	464	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	0.2	$9T18$	0.2	4	$\{27^6, 48^2, \infty\}$	$[7, 7/2]$
$x^9 + t^6x^8 + t^5x^3 + t$	$108 = 2^2 \cdot 3^3$	2	4	600	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	1.7	$9T18$	1.6	4	$\{45^6, 48^2, \infty\}$	$[7, 13/2]$
$x^9 + (t^6 + t^8)x^8 + (t^3 + t^5)x^3 + t$	$108 = 2^2 \cdot 3^3$	2	4	995	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	4.0	$9T18$	3.9	4	$\{27^6, 102^2, \infty\}$	$[16, 7/2]$
$x^{12} + tx + t$	$144 = 2^4 \cdot 3^2$	2	4	290	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.6	$12T84$	0.5	4	$\{6^9, 9^2, \infty\}$	$[1/2, 0]$
$x^{12} + t^2x^{11} + t^5x + t$	$144 = 2^4 \cdot 3^2$	2	4	1118	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	1.2	$12T84$	1.4	4	$\{6^9, 75^2, \infty\}$	$[23/2, 0]$
$x^{12} + (t^3 + t^5)x^5 + t^5x + t$	$144 = 2^4 \cdot 3^2$	2	4	1277	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	4.0	$12T84$	3.8	4	$\{6^9, 93^2, \infty\}$	$[29/2, 0]$
$x^{12} + t^3x^5 + t^5x + t$	$144 = 2^4 \cdot 3^2$	2	4	1277	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	1.7	$12T84$	1.8	4	$\{6^9, 93^2, \infty\}$	$[29/2, 0]$
$x^{12} + tx^9 + t^2x + t$	$144 = 2^4 \cdot 3^2$	2	4	632	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	1.6	$12T84$	1.6	4	$\{6^9, 45^2, \infty\}$	$[13/2, 0]$
$x^{12} + t^5x^5 + t$	$144 = 2^4 \cdot 3^2$	2	4	2593	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	1.2	$12T84$	1.8	4	$\{6^9, 165^2, \infty\}$	$[53/2, 0]$
$x^{12} + t^9x^5 + t$	$144 = 2^4 \cdot 3^2$	2	4	5009	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	5.8	$12T84$	7.7	4	$\{6^9, 309^2, \infty\}$	$[101/2, 0]$
$x^9 + t^2x^8 + tx^5 + tx^3 + t$	$162 = 2 \cdot 3^4$	3	4	318	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	0.3	$9T20$	0.3	4	$\{9^6, 12^2, \infty\}$	$[1, 1/2]$
$x^9 + t^4x^8 + t^3x^5 + tx^3 + t$	$162 = 2 \cdot 3^4$	1	4	575	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	0.9	$9T20$	3.7	4	$\{27^6, 198^2, \infty\}$	$[10, 1/2]$
$x^{27} + t^5x^9 + t^2x^3 + t^3x + t$	$216 = 2^3 \cdot 3^3$	2	3	1307	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^6u_2^4)$	7.3	$27T87$	6.9	3	$\{9^{24}, 54^2, \infty\}$	$[25/2, 5/4]$
$x^{27} + t^7x^9 + t^2x^3 + t^3x + t$	$216 = 2^3 \cdot 3^3$	2	3	1399	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^6u_2^4)$	15.9	$27T87$	15.7	3	$\{9^{24}, 54^2, \infty\}$	$[25/2, 5/4]$
$x^9 + t^2x^8 + tx^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	478	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	1.7	$9T24$	2.4	4	$\{27^6, 144^2, \infty\}$	$[7, 1/2]$
$x^9 + t^5x^7 + t^3x^4 + tx^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	704	$4 = (2) \cdot (2)$	$(2, 2, t = u_2^2)$	5.2	$9T24$	6.3	4	$\{27^6, 189^2, \infty\}$	$[19/2, 1/2]$
$x^9 + t^4x^8 + tx^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	777	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	4.7	$9T24$	5.8	4	$\{27^6, 306^2, \infty\}$	$[16, 1/2]$
$x^9 + (t^2 + t^4)x^8 + (t + t^3)x^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	478	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	1.7	$9T24$	2.2	4	$\{27^6, 144^2, \infty\}$	$[7, 1/2]$
$x^9 + t^5x^7 + t^5x^5 + t^2x^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	870	$4 = (2) \cdot (2)$	$(2, 2, t = u_2^2)$	5.3	$9T24$	9.5	4	$\{54^6, 279^2, \infty\}$	$[29/2, 2]$
$x^9 + t^4x^8 + t^5x^6 + tx^3 + t$	$324 = 2^2 \cdot 3^4$	2	5	929	$4 = (2) \cdot (2)$	$(2, 2, t = 2u_2^2)$	8.1	$9T24$	9.8	4	$\{27^6, 306^2, \infty\}$	$[16, 1/2]$

Continued on next page

Table A.6 – continued from previous page

	$\lvert\mathbf{Spl(f)}\rvert$			**Prec**	**[T:K]**		**T3**		**T4**			
f	$= \lvert\mathbb{F}_{q^l}((u_r))\rvert$	**l**	**r**	(u_r)	$= \mathfrak{f}(T/K)e(T/K)$	**T**	(seq)	**Gal(f)**	(seq)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^9+t^6x^8+t^5x^6+tx^3+t$	$324=2^2\cdot 3^4$	2	5	1290	$4=(2)\cdot(2)$	$(2,2,t=2u_2^2)$	19.8	$9T24$	45.1	4	$\{27^6,468^2,\infty\}$	$[25,1/2]$
$x^9+t^4x^8+t^5x^5+tx^3+t$	$324=2^2\cdot 3^4$	2	5	917	$4=(2)\cdot(2)$	$(2,2,t=2u_2^2)$	8.1	$9T24$	10.0	4	$\{27^6,306^2,\infty\}$	$[16,1/2]$
$x^{27}+tx^{11}+2tx^{10}+2tx^8+tx^7+2tx^6+tx^5+2tx^4+tx^3+2t$	$432=2^4\cdot 3^3$	2	3	429	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	7.9	$27T141$	7.7	2	$\{9^{24},12^2,\infty\}$	$[1/2,1/8]$
$x^{27}+t^2x^9+t^4x^3+t^6x+t$	$486=2\cdot 3^5$	2	3	1812	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	94.2	$27T182$	97.9	6	$\{27^{18},81^6,243^2,\infty\}$	$[26,8,2]$
$x^{12}+t^2x^9+t^5x+t$	$1296=2^4\cdot 3^4$	2	6	5852	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	77.7	$12T212$	170.4	4	$\{54^9,1377^2,\infty\}$	$[49/2,0]$
$x^{12}+tx^9+t^5x+t$	$1296=2^4\cdot 3^4$	2	6	5852	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	239.4	$12T212$	1256.1	4	$\{54^9,1377^2,\infty\}$	$[49/2,0]$
$x^{27}+t^7x^9+t^{13}x^3+t^{15}x+t$	$1458=2\cdot 3^6$	3	4	3303	$2=(1)\cdot(2)$	$(1,1,t=2u_1^2)$	1171.6	$27T347$	1217.4	6	$\{189^{18},486^8,\infty\}$	$[26,19/2]$
$x^{15}+t^4x+t$	$1620=2^2\cdot 3^4\cdot 5$	4	6	3395	$20=(2^2)\cdot(5)$	$(4,2,t=u_2^5)$	12.1	$15T41$	14.0	4	$\{27^{12},648^2,\infty\}$	$[23,0]$
$x^{15}+t^3x^2+t$	$1620=2^2\cdot 3^4\cdot 5$	4	6	2450	$20=(2^2)\cdot(5)$	$(4,2,t=u_2^5)$	9.9	$15T41$	11.0	4	$\{27^{12},459^2,\infty\}$	$[16,0]$
$x^{15}+t^2x+t$	$1620=2^2\cdot 3^4\cdot 5$	4	6	1370	$20=(2^2)\cdot(5)$	$(4,2,t=u_2^5)$	0.4	$15T41$	0.5	3	$\{27^{12},243^2,\infty\}$	$[8,0]$
$x^{27}+t^3x^9+t^2x^3+t^4x+t$	$1944=2^3\cdot 3^5$	2	5	2246	$8=(2)\cdot(2^2)$	$(2,2,t=w^6u_2^4)$	45.2	$27T420$	198.0	4	$\{81^{24},972^2,\infty\}$	$[26,5/4]$
$x^{27}+t^2x^3+t^4x+t$	$1944=2^3\cdot 3^5$	2	5	2246	$8=(2)\cdot(2^2)$	$(2,2,t=w^6u_2^4)$	42.8	$27T420$	199.0	4	$\{81^{24},972^2,\infty\}$	$[26,5/4]$
$x^{27}+t^3x^9+t^4x^3+t^5x+t$	$2916=2^2\cdot 3^6$	6	6	1347	$12=(2\cdot 3)\cdot(2)$	$(6,3,t=u_3^2)$	216.6	$27T480$	262.3	5	$\{81^{24},243^2,\infty\}$	$[25/2,7/2]$
$x^{27}+t^7x^9+t^{10}x^3+t^{11}x+t$	$2916=2^2\cdot 3^6$	6	4	1945	$2=(1)\cdot(2)$	$(1,1,t=2u_1^2)$	1378.5	$27T482$	1403.2	6	$\{189^{18},243^8,\infty\}$	$[25/2,19/2]$
$x^{15}+tx+t$	$3240=2^3\cdot 3^4\cdot 5$	4	6	830	$40=(2^2)\cdot(2\cdot 5)$	$(4,2,t=u_2^{10})$	4.8	$15T52$	5.2	3	$\{54^{12},81^2,\infty\}$	$[1/2,0]$
$x^{15}+t^2x^5+tx+t$	$3240=2^3\cdot 3^4\cdot 5$	4	6	830	$40=(2^2)\cdot(2\cdot 5)$	$(4,2,t=u_2^{10})$	4.5	$15T52$	5.2	3	$\{54^{12},81^2,\infty\}$	$[1/2,0]$
$x^{15}+tx^3+tx+t$	$3240=2^3\cdot 3^4\cdot 5$	4	6	830	$40=(2^2)\cdot(2\cdot 5)$	$(4,2,t=u_2^{10})$	4.4	$15T52$	5.3	3	$\{54^{12},81^2,\infty\}$	$[1/2,0]$
$x^{15}+t^4x^3+tx+t$	$3240=2^3\cdot 3^4\cdot 5$	4	6	830	$40=(2^2)\cdot(2\cdot 5)$	$(4,2,t=u_2^{10})$	7.1	$15T52$	7.5	3	$\{54^{12},81^2,\infty\}$	$[1/2,0]$
$x^{15}+t^3x+t$	$3240=2^3\cdot 3^4\cdot 5$	4	6	4880	$40=(2^2)\cdot(2\cdot 5)$	$(4,2,t=u_2^{10})$	26.0	$15T52$	29.2	3	$\{54^{12},891^2,\infty\}$	$[31/2,0]$
$x^{27}+t^7x^9+t^7x^3+t^9x+t$	$3888=2^4\cdot 3^5$	2	5	6069	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	1334.4	$27T530$	3927.9	4	$\{567^{24},1944^2,\infty\}$	$[26,55/8]$
$x^{24}+tx+t$	$5184=2^6\cdot 3^4$	4	6	1316	$64=(2^2)\cdot(2^4)$	$(4,2,t=2u_2^{16})$	10.1	$24T7775$	16.2	4	$\{54^{21},81^2,\infty\}$	$[1/2,0]$
$x^{18}+tx+t$	$5184=2^6\cdot 3^4$	4	4	452	$64=(2^2)\cdot(2^4)$	$(4,2,t=2u_2^{16})$	216.0	$18T476$	218.0	5	$\{72^9,81^8,\infty\}$	$[1/8,0]$
$x^{18}+tx^{17}+tx^9+t^2x^3+t^3x+t$	$5184=2^6\cdot 3^4$	4	4	1233	$64=(2^2)\cdot(2^4)$	$(4,2,t=2u_2^{16})$	332.2	$18T476$	338.9	5	$\{72^9,225^8,\infty\}$	$[17/8,0]$
$x^{18}+t^3x^{17}+t^3x^9+t^4x^3+t^5x+t$	$5184=2^6\cdot 3^4$	4	4	2375	$64=(2^2)\cdot(2^4)$	$(4,2,t=2u_2^{16})$	1373.3	$18T476$	1380.5	5	$\{72^9,549^8,\infty\}$	$[53/8,0]$
$x^{27}+2tx^{13}+2tx^{12}+tx^{10}+2tx^7+2tx^6+2t$	$5832=2^3\cdot 3^6$	6	6	461	$8=(2)\cdot(2^2)$	$(2,2,t=w^6u_2^4)$	12.5	$27T666$	12.8	3	$\{45^{24},54^2,\infty\}$	$[1/2,1/4]$
$x^{27}+2tx^{20}+tx^{18}+tx^{14}+tx^{12}+2tx^8+2tx^6+2t$	$5832=2^3\cdot 3^6$	6	6	466	$8=(2)\cdot(2^2)$	$(2,2,t=w^2u_2^4)$	9.7	$27T666$	13.1	3	$\{45^{24},72^2,\infty\}$	$[1,1/4]$
$x^{45}+tx+t$	$12960=2^5\cdot 3^4\cdot 5$	4	4	1100	$160=(2^2)\cdot(2^3\cdot 5)$	$(4,2,t=u_2^{40})$	1512.7	$C3^4.C40:C4$	1502.2	5	$\{72^{36},81^8,\infty\}$	$[1/8,0]$

Table A.7: Eisenstein polynomials with one segment over $\mathbb{F}_2((t))$

f	**|Spl(f)|** $= \|\mathbb{F}_{q^l}((u_r))\|$	**l**	**r**	**Prec** (u_r)	**[T:K]** $=\mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	**Gal(f)**	**T4** (sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^2 + (t + t^5 + t^{12})x + t$	$2 = 2$	1	1	200	1	$(1, 0, t = t)$	0.0	C_2	0.0	2	$\{2, \infty\}$	[1]
$x^4 + tx^3 + t$	$8 = 2^3$	2	2	200	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.1	D_4	0.0	3	$\{2^3, \infty\}$	[1]
$x^4 + t^3x^2 + tx + t$	$24 = 2^3 \cdot 3$	2	3	219	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.0	S_4	0.0	4	$\{4^3, \infty\}$	[1/3]
$x^4 + tx^3 + tx^2 + tx + t + t^2$	$24 = 2^3 \cdot 3$	2	3	287	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.1	S_4	0.0	4	$\{4^3, \infty\}$	[1/3]
$x^8 + tx^7 + t$	$24 = 2^3 \cdot 3$	3	2	200	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	0.0	$8T13$	0.0	4	$\{2^7, \infty\}$	[1]
$x^8 + tx^7 + t^{10}x^2 + t^{16}x + t$	$24 = 2^3 \cdot 3$	3	2	322	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	0.1	$8T13$	0.0	4	$\{2^7, \infty\}$	[1]
$x^8 + tx^7 + tx^4 + t$	$56 = 2^3 \cdot 7$	7	2	200	$7 = (7) \cdot (1)$	$(7, 1, t = u_1)$	0.0	$8T25$	0.0	4	$\{2^7, \infty\}$	[1]
$x^8 + tx^7 + tx^6 + tx^5 + tx^4 + tx^3 + tx^2 + tx + t + t^2$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	339	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	1.3	$8T36$	0.0	5	$\{8^7, \infty\}$	[1/7]
$x^8+tx^7+(t+t^2+t^5)x^5+tx^3+(t+t^2)x+t$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	339	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.2	$8T36$	0.0	5	$\{8^7, \infty\}$	[1/7]
$x^8 + tx^7 + t^2x^5 + tx^3 + t$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	353	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.2	$8T36$	0.0	5	$\{10^7, \infty\}$	[3/7]
$x^8 + tx^7 + tx + t$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	339	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.1	$8T36$	0.0	5	$\{8^7, \infty\}$	[1/7]
$x^{16} + t^6x^8 + tx^5 + t^2x + t$	$192 = 2^6 \cdot 3$	4	3	371	$12 = (2^2) \cdot (3)$	$(4, 2, t = u_2^3)$	0.9	$16T433$	0.0	6	$\{4^{15}, \infty\}$	[1/3]
$x^{16} + t^3x + t$	$320 = 2^6 \cdot 5$	4	3	561	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	0.2	$16T711$	0.0	6	$\{16^{15}, \infty\}$	[11/5]
$x^{16} + t^3x^8 + t^2x^5 + t^5x + t$	$320 = 2^6 \cdot 5$	4	3	512	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	1.1	$16T711$	0.0	6	$\{12^{15}, \infty\}$	[7/5]
$x^{16} + tx + t$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	443	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.2	$16T1079$	0.0	6	$\{16^{15}, \infty\}$	[1/15]
$x^{16} + t^2x + t$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	742	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.2	$16T1079$	0.0	6	$\{32^{15}, \infty\}$	[17/15]
$x^{16} + t^2x^7 + t$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	832	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.2	$16T1079$	0.0	6	$\{38^{15}, \infty\}$	[23/15]
$x^{16} + t^2x + t$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	742	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.2	$16T1079$	0.0	6	$\{32^{15}, \infty\}$	[17/15]
$x^{16} + tx^{15} + tx^{14} + tx^{13} + tx^{12} + tx^{11} + tx^{10} + tx^9 + tx^8 + tx^7 + tx^6 + tx^5 + tx^4 + tx^3 + tx^2 + tx + t + t^2$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	443	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	169.4	$16T1079$	0.0	6	$\{16^{15}, \infty\}$	[1/15]
$x^{16} + (t + t^4)x + t$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	443	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.2	$16T1079$	0.0	6	$\{16^{15}, \infty\}$	[1/15]
$x^{16} + t^5x^{15} + t^2x^5 + t^2x + t$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	1778	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	26.0	$16T1079$	0.0	6	$\{32^{15}, \infty\}$	[17/15]
$x^{32} + tx + t$	$4960 = 2^5 \cdot 5 \cdot 31$	5	3	651	$155 = (5) \cdot (31)$	$(5, 2, t = u_2^{31})$	0.8	$C2^5 : (C31 : C5)$	0.0	7	$\{32^{31}, \infty\}$	[1/31]
$x^{32} + t^2x + t$	$4960 = 2^5 \cdot 5 \cdot 31$	5	3	1766	$155 = (5) \cdot (31)$	$(5, 2, t = u_2^{31})$	1.5	$C2^5 : (C31 : C5)$	0.0	7	$\{64^{31}, \infty\}$	[33/31]
$x^{32} + (t + t^3)x + t$	$4960 = 2^5 \cdot 5 \cdot 31$	5	3	651	$155 = (5) \cdot (31)$	$(5, 2, t = u_2^{31})$	1.3	$C2^5 : (C31 : C5)$	0.0	7	$\{32^{31}, \infty\}$	[1/31]
$x^{32} + t^2x^8 + t^2x^5 + t^5x + t$	$4960 = 2^5 \cdot 5 \cdot 31$	5	3	4794	$155 = (5) \cdot (31)$	$(5, 2, t = u_2^{31})$	276.0	$C2^5 : (C31 : C5)$	0.0	7	$\{68^{31}, \infty\}$	[37/31]
$x^{32} + t^5x^{20} + t^2x^9 + t^3x + t$	$4960 = 2^5 \cdot 5 \cdot 31$	5	3	5208	$155 = (5) \cdot (31)$	$(5, 2, t = u_2^{31})$	3201.3	$C2^5 : (C31 : C5)$	0.0	7	$\{72^{31}, \infty\}$	[41/31]
$x^{64} + tx + t$	$24192 = 2^7 \cdot 3^3 \cdot 7$	6	3	1067	$378 = (2{\cdot}3){\cdot}(3^2{\cdot}7)$	$(6, 2, t = u_2^{63})$	9.1	$C2^6.C63.C6$	0.0	8	$\{64^{63}, \infty\}$	[1/63]

Table A.8: Eisenstein polynomials with one segment over $\mathbb{F}_3((t))$

f	**|Spl(f)|** $= \lvert\mathbb{F}_{q^l}((u_r))\rvert$	**l**	**r**	**Prec** (u_r)	**[T:K]** $\mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	**Gal(f)**	**T4** (sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^3 + tx^2 + tx + t + t^2$	$6 = 2 \cdot 3$	1	2	212	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	0.1	S_3	0.0	2	$\{3^2, \infty\}$	[1/2]
$x^3 + (t + t^2)x + t$	$6 = 2 \cdot 3$	1	2	212	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	0.1	S_3	0.0	2	$\{3^2, \infty\}$	[1/2]
$x^3 + t^3x^2 + t^2x + t$	$6 = 2 \cdot 3$	2	2	200	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.1	S_3	0.0	2	$\{3^2, \infty\}$	[2]
$x^3 + (t^3 + t^5)x^2 + (t + t^2)x + t$	$6 = 2 \cdot 3$	1	2	296	$2 = (1) \cdot (2)$	$(1, 1, t = 2u_1^2)$	0.1	S_3	0.0	2	$\{3^2, \infty\}$	[1/2]
$x^9 + tx^2 + t$	$72 = 2^3 \cdot 3^2$	2	3	316	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	0.1	$9T14$	0.0	4	$\{5^8, \infty\}$	[1/4]
$x^9 + t^2x + t$	$72 = 2^3 \cdot 3^2$	2	3	367	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	0.1	$9T14$	0.0	4	$\{9^8, \infty\}$	[5/4]
$x^9 + t^3x^8 + t^6x + t$	$72 = 2^3 \cdot 3^2$	2	3	541	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	0.2	$9T14$	0.0	4	$\{17^8, \infty\}$	[13/4]
$x^9 + t^3x^5 + t^2x + t$	$72 = 2^3 \cdot 3^2$	2	3	367	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	0.1	$9T14$	0.0	4	$\{9^8, \infty\}$	[5/4]
$x^9 + t^5x^7 + t^4x^5 + t^3x^3 + t^2x + t$	$72 = 2^3 \cdot 3^2$	2	3	407	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	0.3	$9T14$	0.0	4	$\{9^8, \infty\}$	[5/4]
$x^9 + tx + t + t^2$	$144 = 2^4 \cdot 3^2$	2	3	352	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.1	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + tx + t$	$144 = 2^4 \cdot 3^2$	2	3	352	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.1	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + tx^8 + tx^7 + tx^6 + tx^5 + tx^4 + tx^3 + tx^2 + tx + t + t^2$	$144 = 2^4 \cdot 3^2$	2	3	352	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	26.3	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + t^3x^7 + tx + t$	$144 = 2^4 \cdot 3^2$	2	3	352	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.2	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + t^3x^3 + tx + t$	$144 = 2^4 \cdot 3^2$	2	3	352	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.1	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + t^5x^7 + t^3x + t$	$144 = 2^4 \cdot 3^2$	2	3	608	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.4	$9T19$	0.0	4	$\{27^8, \infty\}$	[19/8]
$x^9 + (t + t^2)x^5 + tx + t$	$144 = 2^4 \cdot 3^2$	2	3	352	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.2	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + tx^8 + t^3x^7 + tx + t$	$144 = 2^4 \cdot 3^2$	2	3	352	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.3	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + tx^8 + t^6x^3 + tx + t$	$144 = 2^4 \cdot 3^2$	2	3	388	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.3	$9T19$	0.0	4	$\{9^8, \infty\}$	[1/8]
$x^9 + t^5x^8 + t^4x^3 + t^3x + t$	$144 = 2^4 \cdot 3^2$	2	3	608	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.7	$9T19$	0.0	4	$\{27^8, \infty\}$	[19/8]
$x^{27} + tx^{13} + t^3x^3 + t^3x + t$	$162 = 2 \cdot 3^4$	3	3	370	$6 = (3) \cdot (2)$	$(3, 2, t = 2u_2^2)$	3.0	$27T62$	0.0	5	$\{3^{26}, \infty\}$	[1/2]
$x^{27} + tx + t$	$2106 = 2 \cdot 3^4 \cdot 13$	3	3	586	$78 = (3) \cdot (2 \cdot 13)$	$(3, 2, t = 2u_2^{26})$	1.0	$27T422$	0.0	5	$\{27^{26}, \infty\}$	[1/26]
$x^{27} + t^2x + t$	$2106 = 2 \cdot 3^4 \cdot 13$	6	3	871	$78 = (2 \cdot 3) \cdot (13)$	$(6, 2, t = w^{546}u_2^{13})$	4.6	$27T422$	0.0	5	$\{27^{26}, \infty\}$	[14/13]
$x^{27} + tx^{20} + t^3x^3 + t^3x + t$	$2106 = 2 \cdot 3^4 \cdot 13$	6	3	703	$78 = (2 \cdot 3) \cdot (13)$	$(6, 2, t = u_2^{13})$	1170.1	$27T422$	0.0	5	$\{23^{26}, \infty\}$	[10/13]
$x^{27} + t^3x^2 + t^2x + t$	$2106 = 2 \cdot 3^4 \cdot 13$	6	3	1988	$78 = (2 \cdot 3) \cdot (13)$	$(6, 2, t = w^{546}u_2^{13})$	9.0	$27T422$	0.0	5	$\{27^{26}, \infty\}$	[14/13]
$x^{27} + t^3x^3 + t^2x^2 + t$	$2106 = 2 \cdot 3^4 \cdot 13$	3	3	3343	$78 = (3) \cdot (2 \cdot 13)$	$(3, 2, t = 2u_2^{26})$	424.1	$27T422$	0.0	5	$\{55^{26}, \infty\}$	[29/26]
$x^{27} + t^5x^8 + t^4x^3 + t^3x + t$	$2106 = 2 \cdot 3^4 \cdot 13$	3	3	6191	$78 = (3) \cdot (2 \cdot 13)$	$(3, 2, t = 2u_2^{26})$	3075.7	$27T422$	0.0	5	$\{81^{26}, \infty\}$	[55/26]
$x^{27} + t^2x^{26} + t^4x^3 + t^3x + t$	$2106 = 2 \cdot 3^4 \cdot 13$	3	3	5070	$78 = (3) \cdot (2 \cdot 13)$	$(3, 2, t = 2u_2^{26})$	11853.5	$27T422$	0.0	5	$\{79^{26}, \infty\}$	[53/26]
$x^{27} + t^2x^5 + tx + t$	$2106 = 2 \cdot 3^4 \cdot 13$	3	3	786	$78 = (3) \cdot (2 \cdot 13)$	$(3, 2, t = 2u_2^{26})$	162.5	$27T422$	0.0	5	$\{27^{26}, \infty\}$	[1/26]
$x^{81} + tx + t$	$25920 = 2^6 \cdot 3^4 \cdot 5$	4	3	1288	$320 = (2^2) \cdot (2^4 \cdot 5)$	$(4, 2, t = 2u_2^{80})$	45.6	$C3^4.C40.C4.C2$	0.0	6	$\{81^{80}, \infty\}$	[1/80]

Table A.9: Eisenstein polynomials - tamely case over $\mathbb{F}_2((t))$

f	**$\lvert$Spl(f)$\rvert$**	**l**	**r**	**Prec**	**[T:K]**	**T**	**T3**	**Gal(f)**	**T4**	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
	$= \lvert\mathbb{F}_{q^l}((u_r))\rvert$			(u_r)	$= \mathfrak{f}(T/K)e(T/K)$	$(l_T, r_T, t = u_{r_T}^e)$	(sec)		(sec)			
$x^3+tx^2+tx+t+t^2$	$6 = 2 \cdot 3$	2	2	200	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.0	S_3	0.0	2	$\{1^2, \infty\}$	[0]
$x^5+tx^4+tx^3+tx^2+tx+t+t^2$	$20 = 2^2 \cdot 5$	4	2	200	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	0.0	F_5	0.0	2	$\{1^4, \infty\}$	[0]
x^9+tx+t	$54 = 2 \cdot 3^3$	6	2	200	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	0.0	$9T10$	0.0	2	$\{1^8, \infty\}$	[0]
$x^9+t^2x^8+tx^3+t$	$54 = 2 \cdot 3^3$	6	2	200	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	0.0	$9T10$	0.0	2	$\{1^8, \infty\}$	[0]
$x^9+tx^8+tx^7+tx^6+tx^5+tx^4+tx^3+tx^2+tx+t+t^2$	$54 = 2 \cdot 3^3$	6	2	200	$54 = (2 \cdot 3) \cdot (3^2)$	$(6, 2, t = u_2^9)$	0.0	$9T10$	0.0	2	$\{1^8, \infty\}$	[0]
$x^{15}+tx+t$	$60 = 2^2 \cdot 3 \cdot 5$	4	2	200	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.0	$15T6$	0.0	2	$\{1^{14}, \infty\}$	[0]
$x^{15}+t^2x^8+tx^3+t$	$60 = 2^2 \cdot 3 \cdot 5$	4	2	200	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.0	$15T6$	0.0	2	$\{1^{14}, \infty\}$	[0]
$x^{23}+tx+t$	$253 = 11 \cdot 23$	11	2	200	$253 = (11) \cdot (23)$	$(11, 2, t = u_2^{23})$	0.1	$23T3$	0.0	2	$\{1^{22}, \infty\}$	[0]
$x^{27}+tx^{26}+tx^9+t^6x^3+t^3x+t$	$486 = 2 \cdot 3^5$	18	2	200	$486 = (2 \cdot 3^2) \cdot (3^3)$	$(18, 2, t = u_2^{27})$	0.4	$27T176$	0.2	2	$\{1^{26}, \infty\}$	[0]
$x^{27}+tx^{26}+tx^5+t$	$486 = 2 \cdot 3^5$	18	2	200	$486 = (2 \cdot 3^2) \cdot (3^3)$	$(18, 2, t = u_2^{27})$	0.3	$27T176$	0.1	2	$\{1^{26}, \infty\}$	[0]
$x^{27}+tx^{20}+tx^3+tx+t$	$486 = 2 \cdot 3^5$	18	2	200	$486 = (2 \cdot 3^2) \cdot (3^3)$	$(18, 2, t = u_2^{27})$	0.3	$27T176$	0.1	2	$\{1^{26}, \infty\}$	[0]
$x^{27}+tx^{20}+t^3x^3+t^3x+t$	$486 = 2 \cdot 3^5$	18	2	200	$486 = (2 \cdot 3^2) \cdot (3^3)$	$(18, 2, t = u_2^{27})$	0.3	$27T176$	0.1	2	$\{1^{26}, \infty\}$	[0]
$x^{27}+tx^9+tx^3+t$	$486 = 2 \cdot 3^5$	18	2	200	$486 = (2 \cdot 3^2) \cdot (3^3)$	$(18, 2, t = u_2^{27})$	0.1	$27T176$	0.0	2	$\{1^{26}, \infty\}$	[0]
$x^{27}+tx^9+t^3x^3+tx+t$	$486 = 2 \cdot 3^5$	18	2	200	$486 = (2 \cdot 3^2) \cdot (3^3)$	$(18, 2, t = u_2^{27})$	0.2	$27T176$	0.0	2	$\{1^{26}, \infty\}$	[0]
$x^{25}+t^7x^5+t^9x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.1	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+tx+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.2	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+tx^5+tx+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.2	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+tx^5+t^3x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.2	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+tx^5+t^{11}x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.1	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+t^3x^5+t^{11}x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.1	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+t^5x^5+t^7x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.1	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+t^5x^5+t^{13}x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.1	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+t^5x^5+t^{15}x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.1	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25}+t^5x^5+t^{19}x+t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.1	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{43}+t$	$602 = 2 \cdot 7 \cdot 43$	14	2	200	$602 = (2 \cdot 7) \cdot (43)$	$(14, 2, t = u_2^{43})$	0.1	$43T6$	0.0	2	$\{1^{42}, \infty\}$	[0]
$x^{81}+tx+t$	$4374 = 2 \cdot 3^7$	54	2	200	$4374 = (2 \cdot 3^3) \cdot (3^4)$	$(54, 2, t = u_2^{81})$	4.0	$C81.C54$	0.0	2	$\{1^{80}, \infty\}$	[0]
$x^{81}+t^2x^{80}+tx^{27}+tx^9+t^2x^3+t^3x+t$	$4374 = 2 \cdot 3^7$	54	2	200	$4374 = (2 \cdot 3^3) \cdot (3^4)$	$(54, 2, t = u_2^{81})$	4.1	$C81.C54$	1.3	2	$\{1^{80}, \infty\}$	[0]
$x^{125}+tx+t$	$12500 = 2^2 \cdot 5^5$	100	2	200	$12500 = (2^2 \cdot 5^2) \cdot (5^3)$	$(100, 2, t = u_2^{125})$	36.2	$(C25.C25.C5) : C4$	0.3	2	$\{1^{124}, \infty\}$	[0]
$x^{169}+tx+t$	$26364 = 2^2 \cdot 3 \cdot 13^3$	156	2	200	$26364 = (2^2 \cdot 3 \cdot 13) \cdot (13^2)$	$(156, 2, t = u_2^{169})$	125.4	$(C13^2.C13) : C12$	1.1	2	$\{1^{168}, \infty\}$	[0]

Table A.10: Eisenstein polynomials - tamely case over $\mathbb{F}_3((t))$

f	**|Spl(f)|** $= \|\mathbb{F}_{q^l}((u_r))\|$	**l**	**r**	**Prec** (u_r)	**[T:K]** $= \mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	**Gal(f)**	**T4** (sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^4 + t^2x + t$	$8 = 2^3$	2	2	200	$8 = (2) \cdot (2^2)$	$(2, 2, t = u_2^4)$	0.0	D_4	0.0	2	$\{1^3, \infty\}$	[0]
$x^4 + tx^3 + tx^2 + tx + t + t^2$	$8 = 2^3$	2	2	200	$8 = (2) \cdot (2^2)$	$(2, 2, t = u_2^4)$	0.0	D_4	0.0	2	$\{1^3, \infty\}$	[0]
$x^8 + t^2x + t$	$16 = 2^4$	2	2	200	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.0	SD_{16}	0.0	2	$\{1^7, \infty\}$	[0]
$x^8 + t^2x^7 + tx^4 + t$	$16 = 2^4$	2	2	200	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.0	SD_{16}	0.0	2	$\{1^7, \infty\}$	[0]
$x^8 + t^9x^7 + t^4x^6 + tx^4 + t$	$16 = 2^4$	2	2	200	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.1	SD_{16}	0.0	2	$\{1^7, \infty\}$	[0]
$x^8 + t^{15}x^7 + t^{25}x^6 + t^2x^4 + t^4x^2 + t$	$16 = 2^4$	2	2	200	$16 = (2) \cdot (2^3)$	$(2, 2, t = 2u_2^8)$	0.0	SD_{16}	0.0	2	$\{1^7, \infty\}$	[0]
$x^5 + tx + t$	$20 = 2^2 \cdot 5$	4	2	200	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	0.1	F_5	0.0	2	$\{1^4, \infty\}$	[0]
$x^5 + tx^4 + tx^3 + tx^2 + tx + t + t^2$	$20 = 2^2 \cdot 5$	4	2	200	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	0.0	F_5	0.0	2	$\{1^4, \infty\}$	[0]
$x^{10} + tx + t$	$40 = 2^3 \cdot 5$	4	2	200	$40 = (2^2) \cdot (2 \cdot 5)$	$(4, 2, t = u_2^{10})$	0.0	$10T5$	0.0	2	$\{1^9, \infty\}$	[0]
$x^{10} + t^2x^8 + tx^3 + t$	$40 = 2^3 \cdot 5$	4	2	200	$40 = (2^2) \cdot (2 \cdot 5)$	$(4, 2, t = u_2^{10})$	0.0	$10T5$	0.0	2	$\{1^9, \infty\}$	[0]
$x^{16} + t^2x + t$	$64 = 2^6$	4	2	200	$64 = (2^2) \cdot (2^4)$	$(4, 2, t = w^{72}u_2^{16})$	0.0	$16T136$	0.0	2	$\{1^{15}, \infty\}$	[0]
$x^{16} + t^2x^{14} + t^3x^{13} + tx^8 + t$	$64 = 2^6$	4	2	200	$64 = (2^2) \cdot (2^4)$	$(4, 2, t = w^{72}u_2^{16})$	0.2	$16T136$	0.0	2	$\{1^{15}, \infty\}$	[0]
$x^{16} + t^5x^{14} + t^7x^{13} + t^2x^{12} + tx^8 + t$	$64 = 2^6$	4	2	200	$64 = (2^2) \cdot (2^4)$	$(4, 2, t = w^{72}u_2^{16})$	0.2	$16T136$	0.0	2	$\{1^{15}, \infty\}$	[0]
$x^{16} + tx^{15} + tx^{14} + tx^{13} + tx^{12} + tx^{11} + tx^{10} + tx^9 + tx^8 + tx^7 + tx^6 + tx^5 + tx^4 + tx^3 + tx^2 + tx + t + t^2$	$64 = 2^6$	4	2	200	$64 = (2^2) \cdot (2^4)$	$(4, 2, t = w^{72}u_2^{16})$	0.2	$16T136$	0.0	2	$\{1^{15}, \infty\}$	[0]
$x^{14} + tx + t$	$84 = 2^2 \cdot 3 \cdot 7$	6	2	200	$84 = (2 \cdot 3) \cdot (2 \cdot 7)$	$(6, 2, t = u_2^{14})$	0.1	$14T7$	0.0	2	$\{1^{13}, \infty\}$	[0]
$x^{23} + tx + t$	$253 = 11 \cdot 23$	11	2	200	$253 = (11) \cdot (23)$	$(11, 2, t = u_2^{23})$	0.2	$23T3$	0.0	2	$\{1^{22}, \infty\}$	[0]
$x^{32} + tx + t$	$256 = 2^8$	8	2	200	$256 = (2^3) \cdot (2^5)$	$(8, 2, t = w^{6544}u_2^{32})$	0.4	$< 256, 445 >$	0.0	2	$\{1^{31}, \infty\}$	[0]
$x^{32} + t^2x + t$	$256 = 2^8$	8	2	200	$256 = (2^3) \cdot (2^5)$	$(8, 2, t = w^{6544}u_2^{32})$	0.2	$< 256, 445 >$	0.0	2	$\{1^{31}, \infty\}$	[0]
$x^{25} + t^2x + t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.4	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25} + t^3x^5 + t^3x + t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.4	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{25} + t^7x^5 + t^{11}x + t$	$500 = 2^2 \cdot 5^3$	20	2	200	$500 = (2^2 \cdot 5) \cdot (5^2)$	$(20, 2, t = u_2^{25})$	0.2	$25T40$	0.0	2	$\{1^{24}, \infty\}$	[0]
$x^{64} + tx + t$	$1024 = 2^{10}$	16	2	200	$1024 = (2^4) \cdot (2^6)$	$(16, 2, t = u_{(2w)}^{15})$	9.0	$C32.C16.C2$	0.0	2	$\{1^{63}, \infty\}$	[0]
$x^{43} + t$	$1806 = 2 \cdot 3 \cdot 7 \cdot 43$	42	2	200	$1806 = (2 \cdot 3 \cdot 7) \cdot (43)$	$(42, 2, t = u_2^{43})$	0.4	$43T8$	0.0	2	$\{1^{42}, \infty\}$	[0]
$x^{128} + tx + t$	$4096 = 2^{12}$	32	2	200	$4096 = (2^5) \cdot (2^7)$	$(32, 2, t = u_{(2w)}^{31})$	50.5	$C64.C32.C2$	0.4	2	$\{1^{127}, \infty\}$	[0]
$x^{169} + tx + t$	$6591 = 3 \cdot 13^3$	39	2	200	$6591 = (3 \cdot 13) \cdot (13^2)$	$(39, 2, t = u_2^{169})$	143.4	$< 6591, 19 >$	1.2	2	$\{1^{168}, \infty\}$	[0]
$x^{151} + tx + t$	$7550 = 2 \cdot 5^2 \cdot 151$	50	2	200	$7550 = (2 \cdot 5^2) \cdot (151)$	$(50, 2, t = u_2^{151})$	252.9	$< 7550, 3 >$	0.9	2	$\{1^{150}, \infty\}$	[0]
$x^{149} + t^2x + t$	$22052 = 2^2 \cdot 37 \cdot 149$	148	2	200	$22052 = (2^2 \cdot 37) \cdot (149)$	$(148, 2, t = u_2^{149})$	47.3	$< 22052, 3 >$	0.8	2	$\{1^{148}, \infty\}$	[0]
$x^{149} + tx + t$	$22052 = 2^2 \cdot 37 \cdot 149$	148	2	200	$22052 = (2^2 \cdot 37) \cdot (149)$	$(148, 2, t = u_2^{149})$	1077.7	$< 22052, 3 >$	2.6	2	$\{1^{148}, \infty\}$	[0]

Table A.11: Non-Eisenstein irreducible polynomials over $\mathbb{F}_2((t))$

f	**|Spl(f)|** $= \lvert\mathbb{F}_{q^l}((u_r))\rvert$	**l**	**r**	**Prec** (u_r)	**[T:K]** $= \mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	**Gal(f)**	**T4** (sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^4 + tx^3 + (t+t^2)x^2 + t^2x + t^2$	$4 = 2^2$	2	2	209	1	$(1, 0, t = t)$	0.1	C_2^2	0.0	2	$\{1^2, 2, \infty\}$	$[1]$
$x^4 + (t+t^2)x^2 + t^2x + t^2$	$8 = 2^3$	2	3	222	1	$(1, 0, t = t)$	0.2	D_4	0.0	2	$\{2^2, 4, \infty\}$	$[3, 1]$
$x^4 + (t^3+t^9)x + t^2$	$8 = 2^3$	2	2	200	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.1	D_4	0.0	3	$\{4^3, 194\}$	$[3]$
$x^4 + t^3x^2 + t^5x + t^2$	$8 = 2^3$	2	3	1978	1	$(1, 0, t = t)$	2.3	D_4	2.4	3	$\{6^2, 8, \infty\}$	$[7, 5]$
$x^4 + tx^2 + t^2x + t^2$	$8 = 2^3$	2	3	1995	1	$(1, 0, t = t)$	2.4	D_4	2.2	2	$\{2^2, 4, \infty\}$	$[3, 1]$
$x^4 + tx^2 + t^2x + t^2$	$8 = 2^3$	2	3	1995	1	$(1, 0, t = t)$	2.5	D_4	2.5	2	$\{2^2, 4, \infty\}$	$[3, 1]$
$x^4 + t^2x^3 + t^2$	$8 = 2^3$	2	2	200	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.1	D_4	0.0	3	$\{4^3, 194\}$	$[7/3]$
$x^8+t^3x^6+t^4x^5+t^6x^3+(t^6+t^7)x^2+t^7$	$16 = 2^4$	2	2	355	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.2	$8T10$	0.1	3	$\{8^6, 12, 206\}$	$[7, 11/3]$
$x^8 + t^2x^5 + t^2$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	864	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.2	$8T36$	0.2	2	$\{24^7, 830\}$	$[13/7]$
$x^8 + (t+t^2)x^7 + tx^5 + t^2 + t^3$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	339	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	2.0	$8T36$	2.1	2	$\{16^7, 337\}$	$[5/7]$
$x^8 + t^3x + t^2$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	367	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.1	$8T36$	0.1	2	$\{24^7, 207\}$	$[17/7]$
$x^8 + tx^5 + t^3x + t^2$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	339	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	0.8	$8T36$	0.8	2	$\{16^7, 337\}$	$[5/7]$
$x^8 + tx^5 + t^3x + t^2 + t^5$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	339	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	1.1	$8T36$	1.1	2	$\{16^7, 337\}$	$[5/7]$
$x^8+t^2x^7+t^4x^6+(t+t^6)x^5+t^2x^2+t^2$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	381	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	2.2	$8T36$	2.0	2	$\{16^7, 379\}$	$[5/7]$
$x^8+(t^3+t^4)x^7+(t+t^2+t^4+t^6+t^7+t^8+t^9)x^6+(t^2+t^3+t^6+t^9+t^{10})x^5+(t^7+t^8)x^4+(t^2+t^3+t^5+t^6+t^7+t^9+t^{11}+t^{12})x^3+(t^2+t^3+t^4+t^5+t^7+t^8+t^{10}+t^{12}+t^{13})x^2+(t^3+t^6)x+t^2$	$168 = 2^3 \cdot 3 \cdot 7$	3	3	353	$21 = (3) \cdot (7)$	$(3, 2, t = u_2^7)$	2.4	$8T36$	2.2	2	$\{20^7, 335\}$	$[5, 1]$
$x^{16} + t^3x + t^2$	$320 = 2^6 \cdot 5$	4	3	1120	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	1.0	$16T711$	1.1	2	$\{16^{15}, 215\}$	$[11/5]$
$x^{16} + t^2x^9 + t^2$	$320 = 2^6 \cdot 5$	4	3	419	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	4.9	$16T711$	4.7	2	$\{16^{15}, 365\}$	$[5/3]$
$x^{16} + t^2x^9 + t^2 + t^3$	$320 = 2^6 \cdot 5$	4	3	383	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	181.0	$16T711$	180.9	2	$\{16^{15}, 329\}$	$[5/3]$
$x^{16} + t^3x + 1 + t^5$	$320 = 2^6 \cdot 5$	4	3	353	$20 = (2^2) \cdot (5)$	$(4, 2, t = u_2^5)$	33.6	$16T711$	32.9	2	$\{16^{15}, 336\}$	$[11/5]$
$x^{16} + tx^9 + t^3x + t^2$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	443	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	32.7	$16T1079$	32.6	2	$\{32^{15}, 441\}$	$[3/5]$
$x^{16} + t^2x^{11} + t^2$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	1652	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	31.0	$16T1079$	31.1	2	$\{52^{15}, 1490\}$	$[9/5]$
$x^{16} + t^2x + t^2$	$960 = 2^6 \cdot 3 \cdot 5$	4	3	443	$60 = (2^2) \cdot (3 \cdot 5)$	$(4, 2, t = u_2^{15})$	0.3	$16T1079$	0.3	2	$\{32^{15}, 200\}$	$[17/15]$
$x^{32} + t^3x + t^2$	$4960 = 2^5 \cdot 5 \cdot 31$	5	3	3626	$155 = (5) \cdot (31)$	$(5, 2, t = u_2^{31})$	1091.3	$C2^5 : (C31 : C5)$	1091.8	2	$\{96^{31}, 3112\}$	$[65/31]$
$x^{32} + t^2x^9 + t^2$	$4960 = 2^5 \cdot 5 \cdot 31$	5	3	1247	$155 = (5) \cdot (31)$	$(5, 2, t = u_2^{31})$	7177.5	$C2^5 : (C31 : C5)$	7158.8	2	$\{80^{31}, 989\}$	$[41/31]$
$x^{64} + t^2x + 1 + t^3$	$24192 = 2^7 \cdot 3^3 \cdot 7$	6	3	1067	$378 = (2{\cdot}3){\cdot}(3^2{\cdot}7)$	$(6, 2, t = u_2^{63})$	11.8	$C2^6.C63.C6$	14.8	2	$\{128^{63}, 263\}$	$[65/63]$
$x^{64} + tx + 1 + t^2$	$24192 = 2^7 \cdot 3^3 \cdot 7$	6	3	1067	$378 = (2{\cdot}3){\cdot}(3^2{\cdot}7)$	$(6, 2, t = u_2^{63})$	34.7	$C2^6.C63.C6$	34.4	2	$\{64^{63}, 326\}$	$[1/63]$

Table A.12: non-Eisenstein irreducible polynomials over $\mathbb{F}_3((t))$

f	**|Spl(f)|** $= \|\mathbb{F}_{q^l}((u_r))\|$	**l**	**r**	**Prec** (u_r)	**[T:K]** $= \mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	**Gal(f)**	**T4** (sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^3+2t^2x+t^2$	$3=3$	1	1	200	1	$(1,0,u1=u1)$	0.1	C_3	0.0	1	$\{3^2,196\}$	[2]
$x^4+t^2x^2+t^2$	$4=2^2$	2	2	200	$4=(2)\cdot(2)$	$(2,2,t=u_2^2)$	0.1	C_2^2	0.0	2	$\{1^3,\infty\}$	[0]
$x^3+t^2x+t^2$	$6=2\cdot 3$	2	2	200	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	0.1	S_3	0.0	2	$\{3^2,196\}$	[2]
$x^3+t^5x+t^3$	$6=2\cdot 3$	1	2	212	$2=(1)\cdot(2)$	$(1,1,t=2u_1^2)$	0.0	S_3	0.0	2	$\{15^2,204\}$	[13/2]
$x^3+t^3x^2+t^5x+t^3$	$6=2\cdot 3$	2	2	200	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	0.1	S_3	0.0	2	$\{6^2,199\}$	[4]
$x^3+t^3x^2+t^5x+t^2+t^3$	$6=2\cdot 3$	1	2	307	$2=(1)\cdot(2)$	$(1,1,t=2u_1^2)$	0.1	S_3	0.0	2	$\{11^2,287\}$	[4]
$x^3+(t^2+t^5)x+t^2$	$6=2\cdot 3$	2	2	200	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	0.1	S_3	0.0	2	$\{3^2,196\}$	[2]
$x^3+t^2x+1+t^3$	$6=2\cdot 3$	2	2	200	$2=(2)\cdot(1)$	$(2,1,t=u_1)$	0.1	S_3	0.0	2	$\{3^2,196\}$	[2]
$x^8+t^2x^4+t^2$	$8=2^3$	2	2	200	$8=(2)\cdot(2^2)$	$(2,2,t=w^6u_2^4)$	0.2	Q_8	0.0	2	$\{1^7,\infty\}$	[0]
$x^9+(2t^4+2t^6)x^3+t^6x+2t^5$	$9=3^2$	1	1	304	1	$(1,0,t=t)$	0.1	C_3^2	0.0	3	$\{6^6,9^2,200\}$	[8, 5]
$x^9+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$9=3^2$	1	1	242	1	$(1,0,t=t)$	0.4	C_3^2	0.6	3	$\{6^6,15^2,197\}$	[17/4]
$x^9+2t^{18}x^3+t^8x+2t^4$	$36=2^2\cdot 3^2$	4	2	593	$4=(2^2)\cdot(1)$	$(4,1,t=u_1)$	0.2	$9T14$	0.2	2	$\{9^8,203\}$	[8]
$x^{52}+x^{51}+x^{50}+x^{49}+x^{48}+x^{47}+x^{46}+x^{45}+x^{44}+x^{43}+x^{42}+x^{41}+x^{40}+x^{39}+x^{38}+x^{37}+x^{36}+x^{35}+x^{34}+x^{33}+x^{32}+x^{31}+x^{30}+x^{29}+x^{28}+x^{27}+x^{26}+x^{25}+x^{24}+x^{23}+x^{22}+x^{21}+x^{20}+x^{19}+x^{18}+x^{17}+x^{16}+x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^2+x+1+t+t^2+t^3$	$52=2^2\cdot 13$	52	1	200	$52=(2^2\cdot 13)\cdot(1)$	$(52,1,t=u_1)$	23.7	$<52,2>$	0.4	1	$\{0^{51},200\}$	[0]
$x^9+t^2x^5+t^3$	$72=2^3\cdot 3^2$	2	3	316	$8=(2)\cdot(2^2)$	$(2,2,t=u_2^4)$	0.3	$9T16$	0.2	2	$\{15^8,310\}$	[7/4]
$x^9+t^2x+1+t^3$	$72=2^3\cdot 3^2$	2	3	316	$8=(2)\cdot(2^2)$	$(2,2,t=w^6u_2^4)$	4.6	$9T19$	4.5	2	$\{9^8,228\}$	[5/4]
$x^9+t^2x^5+t^3x+t^3$	$144=2^4\cdot 3^2$	2	3	352	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	9.6	$9T19$	8.9	2	$\{27^8,358\}$	[7/4]
$x^9+(t^2+t^3)x^5+(t^3+t^5)x+t^3$	$144=2^4\cdot 3^2$	2	3	352	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	9.5	$9T19$	8.9	2	$\{27^8,358\}$	[7/4]
$x^9+t^2x^5+(t^3+t^5)x+t^3$	$144=2^4\cdot 3^2$	2	3	352	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	9.3	$9T19$	8.5	2	$\{27^8,358\}$	[7/4]
$x^{27}+tx+1+t^2$	$2106=2\cdot 3^4\cdot 13$	3	3	586	$78=(3)\cdot(2\cdot 13)$	$(3,2,t=2u_2^{26})$	2.4	$27T422$	2.4	2	$\{27^{26},252\}$	[1/26]

Table A.13: Reducible polynomials over $\mathbb{F}_2((t))$

f	**|Spl(f)|** $= \|\mathbb{F}_{q^l}((u_r))\|$	**l**	**r**	**Prec** (u_r)	**[T:K]** $= \mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	**Gal(f)**	**T4** (sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^4 + (t+t^2)x^3 + t^3x^2 + (t^2+t^3)x + t^2$	$4 = 2^2$	1	3	220	1	$(1, 0, t = t)$	0.2	$< 4, 2 >$	0.1	2	$\{3^2, 4, \infty\}$	[1]
$x^4 + t^2x^2 + tx + t^{11}$	$6 = 2 \cdot 3$	2	2	1537	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.1	$< 6, 1 >$	0.2	2	$\{1^3, \infty\}$	[1/3]
$x^{17} + (t+t^2+t^3)x + 1 + t + t^2 + t^3$	$8 = 2^3$	8	0	3549	1	$(1, 0, t = t)$	0.9	$< 8, 1 >$	2.3	1	$\{0^{16}, \infty\}$	[0]
$x^{35} + (t+t^2+t^3)x + 1 + t + t^2 + t^3$	$12 = 2^2 \cdot 3$	12	0	5175	1	$(1, 0, t = t)$	3.9	$< 12, 2 >$	8.9	1	$\{0^{34}, \infty\}$	[0]
$x^{16} + t^4x^{14} + t^5x^{13} + t^7x^{11} + t^{10}x^{10} + (t^9 + t^{11})x^9 + (t^8+t^{12})x^8 + t^{11}x^7 + (t^{12}+t^{14}+t^{15})x^6 + (t^{15}+t^{16})x^5 + (t^{16}+t^{17})x^4 + t^{18}x^3 + t^{21}$	$16 = 2^4$	2	1	3773	1	$(1, 0, t = t)$	11.1	$< 16, 3 >$	16.0	3	$\{8^8, 14^6, 18, \infty\}$	[15, 31/7]
$x^{27} + (1+t+t^5)x^3 + t^2x + 1 + t$	$18 = 2 \cdot 3^2$	18	0	7526	1	$(1, 0, t = t)$	16.4	$< 18, 2 >$	25.2	1	$\{0^{26}, \infty\}$	[0]
$x^8 + t^2x^5 + tx + t^2$	$21 = 3 \cdot 7$	3	2	3095	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	0.9	$< 21, 1 >$	2.9	2	$\{1^7, \infty\}$	[1/7]
$x^8 + (t+t^2)x^7 + tx + t^3 + t^5$	$21 = 3 \cdot 7$	3	2	3035	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	0.9	$< 21, 1 >$	2.6	2	$\{1^7, \infty\}$	[1/7]
$x^6 + tx^5 + tx^4 + t^2x^3 + (t+t^3)x^2 + (t^2 + t^3)x + t^2$	$48 = 2^4 \cdot 3$	2	4	61	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.1	$< 48, 48 >$	0.0	3	$\{6^2, 16^3, \infty\}$	[5, 0]
$x^6 + t^2x^5 + tx^4 + tx^3 + (t+t^3)x^2 + (t^2 + t^3)x + t^2$	$48 = 2^4 \cdot 3$	2	4	106	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.1	$< 48, 48 >$	0.2	3	$\{6^2, 8^3, \infty\}$	[3, 0]
$x^{16} + tx + t^2$	$60 = 2^2 \cdot 3 \cdot 5$	4	2	5873	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	0.9	$< 60, 7 >$	11.1	2	$\{1^{15}, \infty\}$	[1/15]
$x^{16} + (t^3+t^5)x^3 + tx + t^3$	$60 = 2^2 \cdot 3 \cdot 5$	4	2	5935	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	5.5	$< 60, 7 >$	18.6	2	$\{1^{15}, \infty\}$	[1/15]
$x^{16} + tx^{11} + (t^3+t^5)x^3 + tx + t^3$	$60 = 2^2 \cdot 3 \cdot 5$	4	2	5935	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	6.4	$< 60, 7 >$	20.0	2	$\{1^{15}, \infty\}$	[1/15]
$x^{16} + tx^{11} + tx + t^2 + t^3$	$60 = 2^2 \cdot 3 \cdot 5$	4	2	5842	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	6.2	$< 60, 7 >$	19.1	2	$\{1^{15}, \infty\}$	[1/15]
$x^{17} + tx + t^2$	$64 = 2^6$	4	2	826	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	0.2	$< 64, 32 >$	1.0	5	$\{1^{16}, \infty\}$	[0]
$x^8 + (t+t^2)x^5 + t^3x^2 + (t^2+t^3)x + t^2$	$96 = 2^5 \cdot 3$	2	4	138	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.1	$< 96, 227 >$	0.6	4	$\{15^4, 16^3, \infty\}$	[5/7]
$x^8 + (t+t^2)x^7 + t^3x^6 + (t+t^2+t^3)x^5 + (t + t^3)x^4 + (t^2+t^3+t^4)x^3 + t^3x^2 + t^4x + t^3$	$96 = 2^5 \cdot 3$	2	5	231	$6 = (2) \cdot (3)$	$(2, 2, t = u_2^3)$	0.2	$< 96, 226 >$	0.8	4	$\{12^4, 16^3, \infty\}$	[5/7]
$x^{32} + tx + t^2$	$155 = 5 \cdot 31$	5	2	9713	$5 = (5) \cdot (1)$	$(5, 1, t = u_1)$	6.9	$< 155, 1 >$	55.0	2	$\{1^{31}, \infty\}$	[1/31]
$x^{32} + tx^{29} + tx + t^2$	$155 = 5 \cdot 31$	5	2	9713	$5 = (5) \cdot (1)$	$(5, 1, t = u_1)$	28.2	$< 155, 1 >$	95.6	2	$\{1^{31}, \infty\}$	[1/31]
$x^{32} + t^2x^{11} + tx + t^2 + t^3$	$155 = 5 \cdot 31$	5	2	10721	$5 = (5) \cdot (1)$	$(5, 1, t = u_1)$	33.7	$< 155, 1 >$	111.5	2	$\{1^{31}, \infty\}$	[1/31]
$x^{32} + t^2x^9 + tx + t^2$	$155 = 5 \cdot 31$	5	2	10280	$5 = (5) \cdot (1)$	$(5, 1, t = u_1)$	24.8	$< 155, 1 >$	95.2	2	$\{1^{31}, \infty\}$	[1/31]
$x^{32} + tx^3 + tx + t^2$	$155 = 5 \cdot 31$	5	2	10847	$5 = (5) \cdot (1)$	$(5, 1, t = u_1)$	25.3	$< 155, 1 >$	103.2	2	$\{1^{31}, \infty\}$	[1/31]
$x^{64} + tx + t^2$	$378 = 2 \cdot 3^3 \cdot 7$	6	2	16075	$6 = (2 \cdot 3) \cdot (1)$	$(6, 1, t = u_1)$	218.9	$< 378, 13 >$	519.3	2	$\{1^{63}, \infty\}$	[1/63]
$x^{16} + t^2x^5 + t^3$	$1100 = 2^2 \cdot 5^2 \cdot 11$	20	4	2202	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	729.8	$< 1100, 27 >$	730.7	2	$\{10^{11}, 11^4, \infty\}$	[7/5]

Table A.14: Reducible polynomials over $\mathbb{F}_3((t))$

	$\lvert$**Spl(f)**$\rvert$			**Prec**	**[T:K]**	**T**	**T3**		**T4**			
f	$= \lvert\mathbb{F}_{q^l}((u_r))\rvert$	**l**	**r**	(u_r)	$= \mathfrak{f}(T/K)e(T/K)$	$(l_T, r_T, t = u_{r_T}^e)$	(sec)	**Gal(f)**	(sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^3 + tx + t^2$	$2 = 2$	1	2	1015	1	$(1, 0, t = t)$	0.1	$< 2, 1 >$	0.1	1	$\{1^2, \infty\}$	[1/2]
$x^{20} + (t + t^2 + t^3)x + 1 + t + t^2 + t^3$	$4 = 2^2$	4	0	1991	1	$(1, 0, t = t)$	0.5	$< 4, 1 >$	1.1	1	$\{0^{19}, \infty\}$	[0]
$x^4 + t^2x^2 + tx + t^{11}$	$6 = 2 \cdot 3$	2	2	1685	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	1.4	$< 6, 1 >$	1.4	2	$\{1^3, \infty\}$	[0]
$x^9 + t^4x^7 + t^8x^5 + t^5x^4 + t^6x$	$8 = 2^3$	2	1	302	1	$(1, 0, t = t)$	0.5	$< 8, 4 >$	0.6	3	$\{3^8, \infty\}$	[17/4]
$x^{35} + (t + t^2 + t^3)x + 1 + t + t^2 + t^3$	$12 = 2^2 \cdot 3$	12	0	5150	1	$(1, 0, t = t)$	37.9	$< 12, 2 >$	48.9	1	$\{0^{34}, \infty\}$	[0]
$x^{17} + (t + t^2 + t^3)x + 1 + t + t^2 + t^3$	$16 = 2^4$	16	0	299	1	$(1, 0, t = t)$	1.5	$< 16, 1 >$	1.8	1	$\{0^{16}, \infty\}$	[0]
$x^9 + t^2x^5 + tx + t^2$	$16 = 2^4$	2	2	3464	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	9.7	$< 16, 8 >$	11.4	3	$\{1^8, \infty\}$	[1/8]
$x^9 + (t + t^2)x^7 + tx + t^3 + t^5$	$16 = 2^4$	2	2	3447	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	10.2	$< 16, 8 >$	11.9	3	$\{1^8, \infty\}$	[1/8]
$x^{38} + (t + t^2 + t^3)x + 1 + t + t^2 + t^3$	$18 = 2 \cdot 3^2$	18	0	7563	1	$(1, 0, t = t)$	328.3	$< 18, 2 >$	385.3	1	$\{0^{37}, \infty\}$	[0]
$x^6 + tx^5 + t^2x^4 + (2t + t^3)x^3 + t^2x^2 + t^3x + t^2$	$18 = 2 \cdot 3^2$	2	3	306	$2 = (2) \cdot (1)$	$(2, 1, t = u_1)$	0.2	$< 18, 4 >$	0.2	3	$\{5^3, 6^2, \infty\}$	[5/2, 0]
$x^{16} + t^4x^{14} + t^5x^{13} + t^7x^{11} + t^{10}x^{10} + (t^9 + t^{11})x^9 + (t^8 + t^{12})x^8 + t^{11}x^7 + (t^{12} + t^{14} + t^{15})x^6 + (t^{15} + t^{16})x^5 + (t^{16} + t^{17})x^4 + t^{18}x^3 + t^{21}$	$32 = 2^5$	4	1	1991	1	$(1, 0, t = t)$	55.4	$< 32, 13 >$	56.5	3	$\{8^{15}, \infty\}$	[0]
$x^{53} + (t + t^2 + t^3)x + 1 + t + t^2 + t^3$	$52 = 2^2 \cdot 13$	52	0	515	1	$(1, 0, t = t)$	95.9	$< 52, 2 >$	115.7	1	$\{0^{52}, \infty\}$	[0]
$x^{17} + tx + t^2$	$64 = 2^6$	4	2	6173	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	53.4	$< 64, 46 >$	63.5	3	$\{1^{16}, \infty\}$	[0]
$x^{27} + tx + t^2$	$78 = 2 \cdot 3 \cdot 13$	3	2	8203	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	115.4	$< 78, 2 >$	149.0	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + tx^{11} + (t^3 + t^5)x^3 + tx + t^3$	$78 = 2 \cdot 3 \cdot 13$	3	2	9369	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	261.7	$< 78, 2 >$	321.4	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + tx^{11} + tx + t^2 + t^3$	$78 = 2 \cdot 3 \cdot 13$	3	2	9422	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	268.2	$< 78, 2 >$	324.0	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + tx^5 + tx + t^3 + t^7$	$78 = 2 \cdot 3 \cdot 13$	3	2	9422	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	225.9	$< 78, 2 >$	279.2	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + tx^5 + tx + t^2$	$78 = 2 \cdot 3 \cdot 13$	3	2	9369	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	238.3	$< 78, 2 >$	291.4	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + t^2x^{11} + tx + t^2$	$78 = 2 \cdot 3 \cdot 13$	3	2	8574	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	216.9	$< 78, 2 >$	260.1	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + t^2x^{11} + tx + t^2 + t^3$	$78 = 2 \cdot 3 \cdot 13$	3	2	9422	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	269.1	$< 78, 2 >$	323.2	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + t^2x^9 + tx + t^2$	$78 = 2 \cdot 3 \cdot 13$	3	2	9157	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	218.4	$< 78, 2 >$	262.3	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + tx^3 + tx + t^5$	$78 = 2 \cdot 3 \cdot 13$	3	2	8256	$3 = (3) \cdot (1)$	$(3, 1, t = u_1)$	159.9	$< 78, 2 >$	201.7	3	$\{1^{26}, \infty\}$	[1/26]
$x^{27} + (1 + t + t^5)x^3 + t^2x + 1 + t$	$108 = 2^2 \cdot 3^3$	4	3	7238	1	$(1, 0, t = t)$	4650.3	$< 108, 37 >$	5011.3	3	$\{0^{24}, 27^2, \infty\}$	[26, −1]
$x^{12} + tx^{10} + tx^9 + tx^5 + (t + t^2)x^3 + t^2x^2 + t^2x + t^2$	$216 = 2^3 \cdot 3^3$	2	4	231	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	0.3	$< 216, 161 >$	7.9	3	$\{12^3, 15^8, \infty\}$	[5/2, 0]
$x^{12} + tx^{10} + tx^9 + t^2x^4 + tx^3 + t^3x^2 + (t^2 + t^3)x + t^2$	$216 = 2^3 \cdot 3^3$	2	4	269	$8 = (2) \cdot (2^2)$	$(2, 2, t = w^2u_2^4)$	0.2	$< 216, 161 >$	0.7	3	$\{12^3, 27^8, \infty\}$	[5, 0]
$x^{27} + t^2x^5 + t^3$	$1100 = 2^2 \cdot 5^2 \cdot 11$	20	4	1135	$4 = (2^2) \cdot (1)$	$(4, 1, t = u_1)$	2389.8	$< 1100, 26 >$	2430.1	2	$\{5^{22}, 11^4, \infty\}$	[16/13]

Table A.15: Strongly Eisenstein polynomials of the form $x^{p^l}+tx+t \in \mathbb{F}_p((t))[x]$

f	**$\|$Spl(f)$\|$** $= \|\mathbb{F}_{q^l}((u_r))\|$	**l**	**r**	**Prec** (u_r)	**[T:K]** $= \mathfrak{f}(T/K)e(T/K)$	**T** $(l_T, r_T, t = u_{r_T}^e)$	**T3** (sec)	**Gal(f)**	**T4** (sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$\mathbf{p=2}$												
x^2+tx+t	$2=2$	1	1	200	1	$(1,0,t=t)$	0.0	C_2	0.0	2	$\{2,\infty\}$	[1]
x^4+tx+t	$24=2^3\cdot 3$	2	3	219	$6=(2)\cdot(3)$	$(2,2,t=u_2^3)$	0.0	S_4	0.0	4	$\{4^3,\infty\}$	[1/3]
x^8+tx+t	$168=2^3\cdot 3\cdot 7$	3	3	339	$21=(3)\cdot(7)$	$(3,2,t=u_2^7)$	0.0	$8T36$	0.0	5	$\{8^7,\infty\}$	[1/7]
$x^{16}+tx+t$	$960=2^6\cdot 3\cdot 5$	4	3	443	$60=(2^2)\cdot(3\cdot 5)$	$(4,2,t=u_2^{15})$	0.2	$16T1079$	0.0	6	$\{16^{15},\infty\}$	[1/15]
$x^{32}+tx+t$	$4960=2^5\cdot 5\cdot 31$	5	3	651	$155=(5)\cdot(31)$	$(5,2,t=u_2^{31})$	0.8	$C2^5:(C31:C5)$	0.0	7	$\{32^{31},\infty\}$	[1/31]
$x^{64}+tx+t$	$24192=2^7\cdot 3^3\cdot 7$	6	3	1067	$378=(2\cdot 3)\cdot(3^2\cdot 7)$	$(6,2,t=u_2^{63})$	8.9	$C2^6.C63.C6$	0.0	8	$\{64^{63},\infty\}$	[1/63]
$x^{128}+tx+t$	$113792=2^7\cdot 7\cdot 127$	7	3	1899	$889=(7)\cdot(127)$	$(7,2,t=u_2^{127})$	163.5	$C2^7.C127.C7$	0.0	9	$\{128^{127},\infty\}$	[1/127]
$x^{256}+tx+t$	$522240=2^{11}\cdot 3\cdot 5\cdot 17$	8	3	3563	$2040=(2^3)\cdot(3\cdot5\cdot17)$	$(8,2,t=u_2^{255})$	4493.9	$C2^8.C255.C8$	0.0	10	$\{256^{255},\infty\}$	[1/255]
$\mathbf{p=3}$												
x^3+tx+t	$6=2\cdot 3$	1	2	212	$2=(1)\cdot(2)$	$(1,1,t=2u_1^2)$	0.0	S_3	0.0	2	$\{3^2,\infty\}$	[1/2]
x^9+tx+t	$144=2^4\cdot 3^2$	2	3	352	$16=(2)\cdot(2^3)$	$(2,2,t=2u_2^8)$	0.0	$9T19$	0.0	4	$\{9^8,\infty\}$	[1/8]
$x^{27}+tx+t$	$2106=2\cdot 3^4\cdot 13$	3	3	586	$78=(3)\cdot(2\cdot 13)$	$(3,2,t=2u_2^{26})$	1.0	$27T422$	0.0	5	$\{27^{26},\infty\}$	[1/26]
$x^{81}+tx+t$	$25920=2^6\cdot 3^4\cdot 5$	4	3	1288	$320=(2^2)\cdot(2^4\cdot 5)$	$(4,2,t=2u_2^{80})$	44.6	$C3^4.C40.C4.C2$	0.0	6	$\{81^{80},\infty\}$	[1/80]
$x^{243}+tx+t$	$294030=2\cdot 3^5\cdot 5\cdot 11^2$	5	3	3394	$1210=(5)\cdot(2\cdot 11^2)$	$(5,2,t=2u_2^{242})$	7333.4	$C3^5.C121.C10$	0.0	7	$\{243^{242},\infty\}$	[1/242]
$\mathbf{p=5}$												
x^5+tx+t	$20=2^2\cdot 5$	1	2	300	$4=(1)\cdot(2^2)$	$(1,1,t=4u_1^4)$	0.0	F_5	0.0	2	$\{5^4,\infty\}$	[1/4]
$x^{25}+tx+t$	$1200=2^4\cdot 3\cdot 5^2$	2	3	560	$48=(2)\cdot(2^3\cdot 3)$	$(2,2,t=4u_2^{24})$	0.9	$25T56$	0.0	4	$\{25^{24},\infty\}$	[1/24]
$x^{125}+tx+t$	$46500=2^2\cdot 3\cdot 5^3\cdot 31$	3	3	1860	$372=(3)\cdot(2^2\cdot 31)$	$(3,2,t=4u_2^{124})$	458.7	$C5^3:(C4*C31:C3)$	0.0	5	$\{125^{124},\infty\}$	[1/124]
$\mathbf{p=7}$												
x^7+tx+t	$42=2\cdot 3\cdot 7$	1	2	326	$6=(1)\cdot(2\cdot 3)$	$(1,1,t=6u_1^6)$	0.1	F_7	0.0	2	$\{7^6,\infty\}$	[1/6]
$x^{49}+tx+t$	$4704=2^5\cdot 3\cdot 7^2$	2	3	872	$96=(2)\cdot(2^4\cdot 3)$	$(2,2,t=6u_2^{48})$	11.1	$C7^2:C8.C6.C2$	0.0	4	$\{49^{48},\infty\}$	[1/48]
$\mathbf{p=11}$												
$x^{11}+tx+t$	$110=2\cdot 5\cdot 11$	1	2	378	$10=(1)\cdot(2\cdot 5)$	$(1,1,t=10u_1^{10})$	0.1	F_{11}	0.0	2	$\{11^{10},\infty\}$	[1/10]
$x^{121}+tx+t$	$29040=2^4\cdot 3\cdot 5\cdot 11^2$	2	3	1808	$240=(2)\cdot(2^3\cdot 3\cdot 5)$	$(2,2,t=10u_2^{120})$	421.8	$C11^2.C60.C2^2$	0.0	4	$\{121^{120},\infty\}$	[1/120]
$\mathbf{p=13}$												
$x^{13}+tx+t$	$156=2^2\cdot 3\cdot 13$	1	2	404	$12=(1)\cdot(2^2\cdot 3)$	$(1,1,t=12u_1^{12})$	0.2	F_{13}	0.0	2	$\{13^{12},\infty\}$	[1/12]
$x^{169}+tx+t$	$56784=2^4\cdot 3\cdot 7\cdot 13^2$	2	3	2432	$336=(2)\cdot(2^3\cdot 3\cdot 7)$	$(2,2,t=12u_2^{168})$	2556.7	$C13^2.C84.C2^2$	0.0	4	$\{169^{168},\infty\}$	[1/168]
$\mathbf{p}=\deg(\mathbf{f})$												
$x^{17}+tx+t$	$272=2^4\cdot 17$	1	2	456	$16=(1)\cdot(2^4)$	$(1,1,t=16u_1^{16})$	0.4	F_{17}	0.0	2	$\{17^{16},\infty\}$	[1/16]
$x^{19}+tx+t$	$342=2\cdot 3^2\cdot 19$	1	2	482	$18=(1)\cdot(2\cdot 3^2)$	$(1,1,t=18u_1^{18})$	0.6	F_{19}	0.0	2	$\{19^{18},\infty\}$	[1/18]
$x^{23}+tx+t$	$506=2\cdot 11\cdot 23$	1	2	534	$22=(1)\cdot(2\cdot 11)$	$(1,1,t=22u_1^{22})$	1.1	F_{23}	0.0	2	$\{23^{22},\infty\}$	[1/22]

Continued on next page

Table A.15 – continued from previous page

	|Spl(f)|			**Prec**	**[T:K]**		**T3**		**T4**			
f	$= \lvert\mathbb{F}_{q^l}((u_r))\rvert$	**l**	**r**	(u_r)	$= \mathfrak{f}(T/K)e(T/K)$	**T**	(sec)	**Gal(f)**	(sec)	$\#\sigma$	$v_{u_r}(\alpha_i - \alpha)$	**Slopes**
$x^{29}+tx+t$	$812 = 2^2 \cdot 7 \cdot 29$	1	2	612	$28 = (1) \cdot (2^2 \cdot 7)$	$(1, 1, t = 28u_1^{28})$	3.5	F_{29}	0.0	2	$\{29^{28}, \infty\}$	[1/28]
$x^{31}+tx+t$	$930 = 2 \cdot 3 \cdot 5 \cdot 31$	1	2	638	$30 = (1) \cdot (2 \cdot 3 \cdot 5)$	$(1, 1, t = 30u_1^{30})$	4.7	$< 930, 1 >$	0.0	2	$\{31^{30}, \infty\}$	[1/30]
$x^{37}+tx+t$	$1332 = 2^2 \cdot 3^2 \cdot 37$	1	2	716	$36 = (1) \cdot (2^2 \cdot 3^2)$	$(1, 1, t = 36u_1^{36})$	10.9	$< 1332, 31 >$	0.0	2	$\{37^{36}, \infty\}$	[1/36]
$x^{41}+tx+t$	$1640 = 2^3 \cdot 5 \cdot 41$	1	2	768	$40 = (1) \cdot (2^3 \cdot 5)$	$(1, 1, t = 40u_1^{40})$	18.0	$< 1640, 47 >$	0.0	2	$\{41^{40}, \infty\}$	[1/40]
$x^{43}+tx+t$	$1806 = 2 \cdot 3 \cdot 7 \cdot 43$	1	2	794	$42 = (1) \cdot (2 \cdot 3 \cdot 7)$	$(1, 1, t = 42u_1^{42})$	23.0	$< 1806, 1 >$	0.0	2	$\{43^{42}, \infty\}$	[1/42]
$x^{47}+tx+t$	$2162 = 2 \cdot 23 \cdot 47$	1	2	846	$46 = (1) \cdot (2 \cdot 23)$	$(1, 1, t = 46u_1^{46})$	36.4	$< 2162, 1 >$	0.0	2	$\{47^{46}, \infty\}$	[1/46]
$x^{53}+tx+t$	$2756 = 2^2 \cdot 13 \cdot 53$	1	2	924	$52 = (1) \cdot (2^2 \cdot 13)$	$(1, 1, t = 52u_1^{52})$	66.0	$< 2756, 3 >$	0.0	2	$\{53^{52}, \infty\}$	[1/52]
$x^{59}+tx+t$	$3422 = 2 \cdot 29 \cdot 59$	1	2	1002	$58 = (1) \cdot (2 \cdot 29)$	$(1, 1, t = 58u_1^{58})$	114.3	$< 3422, 1 >$	0.0	2	$\{59^{58}, \infty\}$	[1/58]
$x^{61}+tx+t$	$3660 = 2^2 \cdot 3 \cdot 5 \cdot 61$	1	2	1028	$60 = (1) \cdot (2^2 \cdot 3 \cdot 5)$	$(1, 1, t = 60u_1^{60})$	135.8	$F61$	0.0	2	$\{61^{60}, \infty\}$	[1/60]
$x^{67}+tx+t$	$4422 = 2 \cdot 3 \cdot 11 \cdot 67$	1	2	1106	$66 = (1) \cdot (2 \cdot 3 \cdot 11)$	$(1, 1, t = 66u_1^{66})$	209.9	$< 4422, 18 >$	0.0	2	$\{67^{66}, \infty\}$	[1/66]
$x^{71}+tx+t$	$4970 = 2 \cdot 5 \cdot 7 \cdot 71$	1	2	1158	$70 = (1) \cdot (2 \cdot 5 \cdot 7)$	$(1, 1, t = 70u_1^{70})$	281.6	$< 4970, 18 >$	0.0	2	$\{71^{70}, \infty\}$	[1/70]
$x^{73}+tx+t$	$5256 = 2^3 \cdot 3^2 \cdot 73$	1	2	1184	$72 = (1) \cdot (2^3 \cdot 3^2)$	$(1, 1, t = 72u_1^{72})$	323.9	$F73$	0.0	2	$\{73^{72}, \infty\}$	[1/72]
$x^{79}+tx+t$	$6162 = 2 \cdot 3 \cdot 13 \cdot 79$	1	2	1262	$78 = (1) \cdot (2 \cdot 3 \cdot 13)$	$(1, 1, t = 78u_1^{78})$	484.6	$< 6162, 30 >$	0.0	2	$\{79^{78}, \infty\}$	[1/78]
$x^{83}+tx+t$	$6806 = 2 \cdot 41 \cdot 83$	1	2	1314	$82 = (1) \cdot (2 \cdot 41)$	$(1, 1, t = 82u_1^{82})$	621.0	$< 6806, 1 >$	0.0	2	$\{83^{82}, \infty\}$	[1/82]
$x^{89}+tx+t$	$7832 = 2^3 \cdot 11 \cdot 89$	1	2	1392	$88 = (1) \cdot (2^3 \cdot 11)$	$(1, 1, t = 88u_1^{88})$	884.9	$F89$	0.0	2	$\{89^{88}, \infty\}$	[1/88]
$x^{97}+tx+t$	$9312 = 2^5 \cdot 3 \cdot 97$	1	2	1496	$96 = (1) \cdot (2^5 \cdot 3)$	$(1, 1, t = 96u_1^{96})$	1362.8	$F97$	0.0	2	$\{97^{96}, \infty\}$	[1/96]
$x^{101}+tx+t$	$10100 = 2^2 \cdot 5^2 \cdot 101$	1	2	1548	$100 = (1) \cdot (2^2 \cdot 5^2)$	$(1, 1, t = 100u_1^{100})$	1566.8	$F101$	0.0	2	$\{101^{100}, \infty\}$	[1/100]
$x^{103}+tx+t$	$10506 = 2 \cdot 3 \cdot 17 \cdot 103$	1	2	1574	$102 = (1) \cdot (2 \cdot 3 \cdot 17)$	$(1, 1, t = 102u_1^{102})$	1814.8	$< 10506, 18 >$	0.0	2	$\{103^{102}, \infty\}$	[1/102]
$x^{107}+tx+t$	$11342 = 2 \cdot 53 \cdot 107$	1	2	1626	$106 = (1) \cdot (2 \cdot 53)$	$(1, 1, t = 106u_1^{106})$	2195.9	$< 11342, 1 >$	0.0	2	$\{107^{106}, \infty\}$	[1/106]
$x^{109}+tx+t$	$11772 = 2^2 \cdot 3^3 \cdot 109$	1	2	1652	$108 = (1) \cdot (2^2 \cdot 3^3)$	$(1, 1, t = 108u_1^{108})$	2448.7	$F109$	0.0	2	$\{109^{108}, \infty\}$	[1/108]
$x^{113}+tx+t$	$12656 = 2^4 \cdot 7 \cdot 113$	1	2	1704	$112 = (1) \cdot (2^4 \cdot 7)$	$(1, 1, t = 112u_1^{112})$	2852.3	$F113$	0.0	2	$\{113^{112}, \infty\}$	[1/112]
$x^{127}+tx+t$	$16002 = 2 \cdot 3^2 \cdot 7 \cdot 127$	1	2	1886	$126 = (1) \cdot (2 \cdot 3^2 \cdot 7)$	$(1, 1, t = 126u_1^{126})$	4986.0	$F127$	0.0	2	$\{127^{126}, \infty\}$	[1/126]
$x^{131}+tx+t$	$17030 = 2 \cdot 5 \cdot 13 \cdot 131$	1	2	1938	$130 = (1) \cdot (2 \cdot 5 \cdot 13)$	$(1, 1, t = 130u_1^{130})$	5865.5	$< 17030, 18 >$	0.0	2	$\{131^{130}, \infty\}$	[1/130]
$x^{137}+tx+t$	$18632 = 2^3 \cdot 17 \cdot 137$	1	2	2016	$136 = (1) \cdot (2^3 \cdot 17)$	$(1, 1, t = 136u_1^{136})$	7310.9	$F137$	0.0	2	$\{137^{136}, \infty\}$	[1/136]
$x^{139}+tx+t$	$19182 = 2 \cdot 3 \cdot 23 \cdot 139$	1	2	2042	$138 = (1) \cdot (2 \cdot 3 \cdot 23)$	$(1, 1, t = 138u_1^{138})$	7929.9	$< 19182, 18 >$	0.0	2	$\{139^{138}, \infty\}$	[1/138]
$x^{149}+tx+t$	$22052 = 2^2 \cdot 37 \cdot 149$	1	2	2172	$148 = (1) \cdot (2^2 \cdot 37)$	$(1, 1, t = 148u_1^{148})$	10988.6	$< 22052, 3 >$	0.0	2	$\{149^{148}, \infty\}$	[1/148]
$x^{151}+tx+t$	$22650 = 2 \cdot 3 \cdot 5^2 \cdot 151$	1	2	2198	$150 = (1) \cdot (2 \cdot 3 \cdot 5^2)$	$(1, 1, t = 150u_1^{150})$	11758.0	$F151$	0.0	2	$\{151^{150}, \infty\}$	[1/150]

A.4 Factorization Code

Let us first describe the notation.
Let $f \in K[X]$, where $K := \mathbb{F}_q((t))$.

- Prec(t): The t-precision of the base field K.
- time1(seq): The time needed for the factorization of $f \in K[X]$ using Magma's code.
- time2(seq): The time needed for the factorization of $f \in K[X]$ employing the new factorization code.

We remark that we factorize the polynomials by using the "Extension"-parameter in order to check the running time required for the computation of the defining polynomial of the corresponding extensions.

Table A.16: Factorization code examples

f	**q**	**Prec** (t)	**time1** (sec)	**time2** (sec)
$x^{27} + tx^{24} + t^2x^9 + t^2$	$2 = 2$	2108	2.72	1.17
$x^9 + t^4x^7 + t^8x^5 + t^5x^4 + t^6x$	$2 = 2$	2108	0.25	0.18
$x^{81} + tx + t$	$2 = 2$	2108	0.06	0.06
$x^{17} + tx + t^2$	$2 = 2$	2108	0.20	0.15
$x^8 + (t^{10} + t^{12} + t^{15} + t^{18})x^4 + (t^{17} + t^{22} + t^{23} + t^{28} + t^{30})x^2 + (t^{22} + t^{25} + t^{28} + t^{30})x + t^{10} + t^{17} + t^{18} + t^{19} + t^{20} + t^{22} + t^{23} + t^{28} + t^{31} + t^{35} + t^{36} + t^{38} + t^{40}$	$2 = 2$	2108	3.09	1.34
$x^8 + (t^{10} + t^{12} + t^{15} + t^{18})x^4 + (t^{17} + t^{22} + t^{23} + t^{28} + t^{30})x^2 + (t^{22} + t^{25} + t^{28} + t^{30})x + t^{10} + t^{17} + t^{18} + t^{19} + t^{20} + t^{22} + t^{23} + t^{28} + t^{31} + t^{35} + t^{36} + t^{38} + t^{40}$	$2 = 2$	5000	13.35	3.65
$x^{27} + tx + t$	$4 = 2^2$	21000	0.11	0.11
$x^{27} + tx^{24} + t^2x^9 + t^2$	$4 = 2^2$	21000	142.53	9.14
$x^9 + t^4x^7 + t^8x^5 + t^5x^4 + t^6x$	$4 = 2^2$	21000	5.56	1.43
$x^{27} + (1 + t + t^5)x^3 + t^2x + 1 + t$	$4 = 2^2$	21000	55.06	36.68
$x^{30} + (1 + t + t^5)x^3 + t^2x + 1 + t$	$4 = 2^2$	21000	45.61	25.67
$x^{53} + (t + t^2 + t^3)x + 1 + t + t^2 + t^3$	$4 = 2^2$	21000	297.24	250.00
$x^{17} + tx + t^2$	$4 = 2^2$	21000	1.47	0.71
$x^6 + x^4 + x^2$	$4 = 2^2$	21000	0.02	0.02
$x^9 + tx + t$	$4 = 2^2$	21000	0.04	0.07
$x^8 + (t^{10} + t^{12} + t^{15} + t^{18})x^4 + (t^{17} + t^{22} + t^{23} + t^{28} + t^{30})x^2 + (t^{22} + t^{25} + t^{28} + t^{30})x + t^{10} + t^{17} + t^{18} + t^{19} + t^{20} + t^{22} + t^{23} + t^{28} + t^{31} + t^{35} + t^{36} + t^{38} + t^{40}$	$4 = 2^2$	21000	201.93	22.11
$x^{54} + (t + t^2 + t^3)x + 1 + t + t^2 + t^3$	$4 = 2^2$	21000	5963.45	5809.15
$x^8 + b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$*	$4 = 2^2$	21000	282.14	95.68
$x^2 + a_1x + a_0$ †	$4 = 2^2$	26880	41.22	0.29
$x^{27} + tx + t$	$3 = 3$	2000	0.03	0.03
$x^9 + t^4x^7 + t^8x^5 + t^5x^4 + (2t^4 + 2t^{10})x^3 + t^6x + 2t^3$	$3 = 3$	2000	1.25	0.54
$x^{27} + tx^{11} + 2tx^{10} + 2tx^8 + tx^7 + 2tx^6 + tx^5 + 2tx^4 + tx^3 + 2t$	$3 = 3$	2000	0.23	0.23
$x^{27} + t^4x^7 + t^8x^5 + t^5x^4 + (2t^4 + 2t^{10})x^3 + t^6x + 2t^3$	$3 = 3$	2000	47.92	18.24
$x^{36} + t^4x^7 + t^8x^5 + t^5x^4 + (2t^4 + 2t^{10})x^3 + t^6x + 2t^3$	$3 = 3$	2000	144.53	51.39
$x^{36} + 2tx^{20} + tx^{18} + tx^{14} + tx^{12} + 2tx^8 + 2tx^6 + 2t$	$3 = 3$	2000	0.35	0.33
$x^{17} + tx + t^2$	$3 = 3$	2000	0.25	0.23

Continued on next page

*See f_2 after the end of the table, p. 128.

†See f_3 in p. 129.

Table A.16 – continued from previous page

f	**q**	**Prec** (t)	**time1** (sec)	**time2** (sec)
$x^8+(t^{10}+t^{12}+t^{15}+t^{18})x^4+(t^{17}+t^{22}+t^{23}+t^{28}+t^{30})x^2+(t^{22}+t^{25}+t^{28}+t^{30})x+t^{10}+t^{17}+t^{18}+t^{19}+t^{20}+t^{22}+t^{23}+t^{28}+t^{31}+t^{35}+t^{36}+t^{38}+t^{40}$	$3=3$	2000	1.65	0.92
$x^4+t^3x+t^2+t^3+t^4$	$3=3$	2000	0.10	0.09
$x^{10}+t^2x^8+tx^3+t$	$3=3$	5000	0.22	0.21
$x^9+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$3=3$	5000	5.11	1.71
$x^{27}+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$3=3$	5000	180.43	65.56
$x^{17}+tx+t^2$	$3=3$	5000	0.67	0.58
$x^8+(t^{10}+t^{12}+t^{15}+t^{18})x^4+(t^{17}+t^{22}+t^{23}+t^{28}+t^{30})x^2+(t^{22}+t^{25}+t^{28}+t^{30})x+t^{10}+t^{17}+t^{18}+t^{19}+t^{20}+t^{22}+t^{23}+t^{28}+t^{31}+t^{35}+t^{36}+t^{38}+t^{40}$	$3=3$	5000	5.72	3.26
$x^4+t^3x+t^2+t^3+t^4$	$3=3$	5000	0.31	0.15
$x^3+c_2x^2+c_1x+c_0$ [‡]	$3=3$	5000	0.15	0.07
$x^9+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$9=3^2$	2000	2.56	1.24
$x^{27}+(1+t+t^5)x^3+t^2x+1+t$	$9=3^2$	2000	18.31	18.26
$x^{27}+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$9=3^2$	2000	197.79	54.71
$x^{36}+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$9=3^2$	2000	950.75	204.84
$x^{17}+tx+t^2$	$9=3^2$	2000	0.39	0.37
$x^8+(t^{10}+t^{12}+t^{15}+t^{18})x^4+(t^{17}+t^{22}+t^{23}+t^{28}+t^{30})x^2+(t^{22}+t^{25}+t^{28}+t^{30})x+t^{10}+t^{17}+t^{18}+t^{19}+t^{20}+t^{22}+t^{23}+t^{28}+t^{31}+t^{35}+t^{36}+t^{38}+t^{40}$	$9=3^2$	2000	1.64	0.98
$x^{18}+(2t^8+t^{12})x^{12}+t^{12}x^{10}+(t^{16}+t^{20}+t^{24})x^6+(t^{20}+2t^{24})x^4+t^{24}x^2+t^{12}+2t^{16}+t^{20}$	$9=3^2$	2000	2.86	2.85
$x^9+tx+2t+2t^2$	$9=3^2$	5000	0.03	0.03
$x^9+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$9=3^2$	5000	13.60	5.04
$x^{27}+(1+t+t^5)x^3+t^2x+1+t$	$9=3^2$	5000	56.53	52.80
$x^3+c_2x^2+c_1x+c_0$ [§]	$9=3^2$	5000	0.26	0.09
$x^8+(t^{10}+t^{12}+t^{15}+t^{18})x^4+(t^{17}+t^{22}+t^{23}+t^{28}+t^{30})x^2+(t^{22}+t^{25}+t^{28}+t^{30})x+t^{10}+t^{17}+t^{18}+t^{19}+t^{20}+t^{22}+t^{23}+t^{28}+t^{31}+t^{35}+t^{36}+t^{38}+t^{40}$	$9=3^2$	5000	5.13	3.09
$x^4+t^3x+t^2+t^3+t^4$	$9=3^2$	5000	0.250	0.240
$x^{18}+(2t^8+t^{12})x^{12}+t^{12}x^{10}+(t^{16}+t^{20}+t^{24})x^6+(t^{20}+2t^{24})x^4+t^{24}x^2+t^{12}+2t^{16}+t^{20}$	$9=3^2$	5000	21.450	15.680
$x^{36}+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$9=3^2$	5000	3187.62	596.32
$x^{45}+t^4x^7+t^8x^5+t^5x^4+(2t^4+2t^{10})x^3+t^6x+2t^3$	$9=3^2$	5000	7279.95	1290.78

[‡] See f_1 in p. 128.

[§] See f_1 in p. 128.

Now we give the polynomials f_1, f_2 and f_3 used in the above table.

$$f_1 := x^3 + c_2x^2 + c_1x + c_0 = x^3 + (t^2 + 2t^3 + t^4 + 2t^5 + t^6 + t^7 + 2t^9 + 2t^{10} + 2t^{12} + t^{13} +
t^{15} + 2t^{16} + 2t^{17} + 2t^{20} + 2t^{21} + 2t^{22} + t^{23} + 2t^{24} + 2t^{25} + t^{26} + t^{27} + 2t^{30} + t^{34} + 2t^{37} + 2t^{39} + t^{40} +
2t^{41} + 2t^{44} + 2t^{45} + t^{47} + t^{48} + t^{50} + t^{52} + 2t^{53} + 2t^{55} + t^{57} + 2t^{59} + t^{60} + t^{61} + t^{62} + 2t^{63} + 2t^{66} + t^{68} + t^{69} +
2t^{70} + 2t^{72} + t^{73} + t^{74} + 2t^{75} + t^{76} + 2t^{77} + 2t^{78} + 2t^{79} + t^{80} + t^{81} + t^{83} + 2t^{85} + t^{86} + t^{87} + t^{88} + 2t^{89} + 2t^{90} +
2t^{93} + 2t^{94} + 2t^{95} + t^{96} + 2t^{97} + t^{98} + 2t^{99} + t^{101} + t^{102} + 2t^{103} + 2t^{104} + 2t^{105} + t^{106} + t^{109} + t^{110} + t^{111} +
2t^{112} + t^{113} + 2t^{114} + 2t^{116} + 2t^{117} + t^{119} + 2t^{120} + 2t^{121} + 2t^{124} + 2t^{125} + 2t^{126} + t^{127} + t^{128} + 2t^{129} +
2t^{131} + 2t^{132} + 2t^{133} + 2t^{135} + 2t^{136} + t^{137} + 2t^{142} + t^{143} + 2t^{144} + t^{145} + 2t^{146} + 2t^{147} + 2t^{149})x + 2 + 2t +
t^2 + 2t^3 + t^4 + t^6 + t^7 + 2t^{11} + t^{16} + 2t^{17} + 2t^{18} + t^{19} + 2t^{20} + 2t^{21} + 2t^{22} + 2t^{23} + 2t^{24} + 2t^{27} + 2t^{29} + 2t^{30} +
2t^{31} + t^{32} + t^{33} + t^{34} + 2t^{37} + 2t^{39} + 2t^{41} + t^{42} + 2t^{43} + t^{44} + 2t^{47} + t^{48} + t^{49} + t^{50} + 2t^{51} + 2t^{52} + t^{54} + 2t^{57} +
t^{58} + t^{59} + t^{60} + t^{61} + t^{62} + t^{63} + t^{64} + t^{65} + t^{66} + 2t^{70} + t^{71} + 2t^{72} + 2t^{73} + 2t^{74} + 2t^{75} + t^{77} + t^{78} + t^{80} + t^{81} +
2t^{82} + 2t^{83} + 2t^{84} + t^{85} + t^{87} + t^{89} + t^{90} + 2t^{91} + 2t^{92} + t^{93} + t^{95} + 2t^{97} + 2t^{98} + t^{100} + 2t^{101} + t^{102} + 2t^{103} +
2t^{104} + t^{105} + 2t^{106} + 2t^{108} + 2t^{110} + 2t^{111} + t^{114} + t^{115} + 2t^{116} + 2t^{117} + 2t^{118} + t^{119} + 2t^{120} + 2t^{121} +
2t^{124} + 2t^{128} + 2t^{130} + t^{131} + 2t^{132} + t^{133} + 2t^{134} + 2t^{135} + 2t^{136} + 2t^{139} + 2t^{141} + t^{142} + t^{143} + t^{144} + t^{146} + 2t^{147}$$

$$f_2 := x^8 + b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = x^8 + (t^{88} + t^{92} + t^{104} + t^{108} + t^{118} + t^{122} +
t^{148} + t^{156} + t^{164} + t^{172} + t^{175} + t^{178} + t^{180} + t^{183} + t^{184} + t^{186})x^7 + (t^{32} + t^{48} + t^{64} + t^{80} + t^{96} + t^{112} + t^{128} +
t^{144} + t^{152} + t^{160} + t^{168} + t^{176} + t^{184} + t^{192} + t^{200} + t^{208} + t^{224} + t^{240} + t^{256} + t^{272} + t^{280} + t^{288} + t^{296} + t^{304} +
t^{312} + t^{320} + t^{328} + t^{336} + t^{352} + t^{368} + t^{380} + t^{384} + t^{400} + t^{408} + t^{416} + t^{424} + t^{432} + t^{440} + t^{444} + t^{448} + t^{456} +
t^{464} + t^{480} + t^{496} + t^{500} + t^{512} + t^{528} + t^{536} + t^{544} + t^{552} + t^{560} + t^{568} + t^{576} + t^{584} + t^{592} + t^{608} + t^{620} + t^{624} +
t^{632} + t^{636} + t^{640} + t^{648} + t^{656} + t^{672} + t^{684} + t^{688} + t^{696} + t^{700} + t^{704} + t^{712} + t^{720} + t^{728} + t^{736} + t^{740} + t^{744} +
t^{748} + t^{752} + t^{756} + t^{760} + t^{768} + t^{776} + t^{784} + t^{800} + t^{812} + t^{816} + t^{824} + t^{832} + t^{840} + t^{856} + t^{868} + t^{872} + t^{880} +
t^{892} + t^{896} + t^{920} + t^{936} + t^{944} + t^{952} + t^{956} + t^{960} + t^{968} + t^{976} + t^{980} + t^{992} + t^{1008} + t^{1012} + t^{1024} + t^{1040} +
t^{1048} + t^{1056} + t^{1064} + t^{1072} + t^{1080} + t^{1088} + t^{1096} + t^{1104} + t^{1120} + t^{1132} + t^{1136} + t^{1144} + t^{1148} + t^{1152} + t^{1160} +
t^{1168})x^6 + (t^8 + t^{12} + t^{16} + t^{20} + t^{24} + t^{28} + t^{32} + t^{36} + t^{38} + t^{40} + t^{42} + t^{44} + t^{46} + t^{48} + t^{50} + t^{52} + t^{56} + t^{60} +
t^{64} + t^{68} + t^{70} + t^{72} + t^{74} + t^{76} + t^{78} + t^{80} + t^{82} + t^{84} + t^{88} + t^{92} + t^{95} + t^{96} + t^{100} + t^{102} + t^{104} + t^{106} + t^{108} +
t^{110} + t^{111} + t^{112} + t^{114} + t^{116} + t^{120} + t^{124} + t^{125} + t^{128} + t^{132} + t^{134} + t^{136} + t^{138} + t^{140} + t^{142} + t^{144} + t^{146} +
t^{148} + t^{152} + t^{155} + t^{156} + t^{158} + t^{159} + t^{160} + t^{162} + t^{164} + t^{168} + t^{171} + t^{172} + t^{174} + t^{175} + t^{176} + t^{178} + t^{180} +
t^{182} + t^{184} + t^{185} + t^{186} + t^{187} + t^{188} + t^{189} + t^{190} + t^{192} + t^{194} + t^{196} + t^{200} + t^{203} + t^{204} + t^{206} + t^{208} + t^{210} +
t^{214} + t^{217} + t^{218} + t^{220} + t^{223} + t^{224} + t^{230} + t^{234} + t^{236} + t^{238} + t^{239} + t^{240} + t^{242} + t^{244} + t^{245} + t^{248} + t^{252} +
t^{253} + t^{256} + t^{260} + t^{262} + t^{264} + t^{266} + t^{268} + t^{270} + t^{272} + t^{274} + t^{276} + t^{280} + t^{283} + t^{284} + t^{286} + t^{287} + t^{288} +
t^{290} + t^{292} + t^{296} + t^{299} + t^{300} + t^{303} + t^{304} + t^{305} + t^{308} + t^{310} + t^{312} + t^{313} + t^{314} + t^{315} + t^{316} + t^{317} + t^{318} +
t^{320} + t^{322} + t^{324} + t^{328} + t^{331} + t^{332} + t^{335} + t^{336} + t^{337} + t^{342} + t^{345} + t^{346} + t^{348} + t^{352} + t^{356} + t^{358} + t^{360} +
t^{362} + t^{365} + t^{367} + t^{369} + t^{372} + t^{373} + t^{376} + t^{380} + t^{381} + t^{384} + t^{388} + t^{390} + t^{392} + t^{394} + t^{396} + t^{399} + t^{400} +
t^{401} + t^{404} + t^{408} + t^{411} + t^{412} + t^{414} + t^{416} + t^{418} + t^{427} + t^{429} + t^{430} + t^{431} + t^{434} + t^{436} + t^{438} + t^{440} + t^{441} +
t^{442} + t^{444} + t^{445} + t^{446} + t^{448} + t^{450} + t^{459} + t^{460} + t^{462} + t^{464} + t^{466} + t^{468} + t^{470} + t^{472} + t^{473} + t^{474} + t^{475} +
t^{476} + t^{479} + t^{480} + t^{485} + t^{486} + t^{490} + t^{492} + t^{494} + t^{495} + t^{496} + t^{498} + t^{500} + t^{501} + t^{503} + t^{504} + t^{508} + t^{509} +
t^{512} + t^{516} + t^{518} + t^{520} + t^{522} + t^{524} + t^{526} + t^{528} + t^{530} + t^{532} + t^{536} + t^{539} + t^{540} + t^{542} + t^{543} + t^{544} + t^{546} +
t^{548} + t^{552} + t^{555} + t^{556} + t^{559} + t^{561} + t^{566} + t^{568} + t^{569} + t^{570} + t^{571} + t^{572} + t^{573} + t^{574} + t^{576} + t^{578} + t^{580} +
t^{584} + t^{587} + t^{588} + t^{591} + t^{593} + t^{598} + t^{600} + t^{601} + t^{602} + t^{612} + t^{614} + t^{616} + t^{618} + t^{621} + t^{623} + t^{624} + t^{625} +
t^{629} + t^{632} + t^{635} + t^{636} + t^{637} + t^{638} + t^{640} + t^{642} + t^{644} + t^{648} + t^{651} + t^{652} + t^{655} + t^{657} + t^{660} + t^{662} + t^{665} +
t^{666} + t^{678} + t^{682} + t^{685} + t^{686} + t^{687} + t^{688} + t^{690} + t^{692} + t^{700} + t^{701} + t^{702} + t^{706} + t^{715} + t^{718} + t^{720} + t^{722} +
t^{725} + t^{726} + t^{728} + t^{729} + t^{730} + t^{731} + t^{732} + t^{734} + t^{735} + t^{738} + t^{741} + t^{743} + t^{748} + t^{750} + t^{751} + t^{754} + t^{757} +
t^{759} + t^{760} + t^{763} + t^{764} + t^{765} + t^{766} + t^{768} + t^{770} + t^{772} + t^{776} + t^{779} + t^{780} + t^{782} + t^{786} + t^{788} + t^{790} + t^{793} +
t^{794} + t^{799} + t^{804} + t^{806} + t^{808} + t^{810} + t^{812} + t^{815} + t^{816} + t^{817} + t^{827} + t^{828} + t^{829} + t^{830} + t^{834} + t^{836} + t^{840} +
t^{843} + t^{847} + t^{849} + t^{851} + t^{852} + t^{853} + t^{856} + t^{857} + t^{864} + t^{867} + t^{868} + t^{871} + t^{872} + t^{877} + t^{878} + t^{879} + t^{882} +
t^{884} + t^{885} + t^{888} + t^{892} + t^{893} + t^{896} + t^{900} + t^{902} + t^{904} + t^{906} + t^{908} + t^{911} + t^{912} + t^{913} + t^{915} + t^{916} + t^{918} +
t^{920} + t^{922} + t^{924} + t^{928} + t^{931} + t^{932} + t^{936} + t^{939} + t^{941} + t^{943} + t^{945} + t^{950} + t^{953} + t^{954} + t^{955} + t^{956} + t^{957} +
t^{958} + t^{960} + t^{962} + t^{965} + t^{971} + t^{972} + t^{974} + t^{976} + t^{978} + t^{980} + t^{982} + t^{983} + t^{984} + t^{985} + t^{986} + t^{987} + t^{988} +
t^{991} + t^{992} + t^{997} + t^{1001} + t^{1004} + t^{1006} + t^{1007} + t^{1008} + t^{1010} + t^{1012} + t^{1013} + t^{1015} + t^{1016} + t^{1020} + t^{1021} +
t^{1024} + t^{1028} + t^{1030} + t^{1032} + t^{1034} + t^{1036} + t^{1038} + t^{1040} + t^{1042} + t^{1044} + t^{1048} + t^{1051} + t^{1052} + t^{1054} +
t^{1055} + t^{1056} + t^{1058} + t^{1060} + t^{1064} + t^{1067} + t^{1068} + t^{1071} + t^{1073} + t^{1078} + t^{1080} + t^{1081} + t^{1082} + t^{1083} + t^{1084} +
t^{1085} + t^{1086} + t^{1088} + t^{1090} + t^{1092} + t^{1096} + t^{1099} + t^{1100} + t^{1103} + t^{1105} + t^{1110} + t^{1112} + t^{1113} + t^{1114} + t^{1124} +
t^{1126} + t^{1128} + t^{1130} + t^{1133} + t^{1135} + t^{1136} + t^{1137} + t^{1139} + t^{1141} + t^{1144} + t^{1147} + t^{1148} + t^{1149} + t^{1150} + t^{1152} +
t^{1154} + t^{1156} + t^{1160} + t^{1163} + t^{1164} + t^{1167} + t^{1169})x^4 + t^8 + t^{12} + t^{16} + t^{20} + t^{24} + t^{28} + t^{32} + t^{36} + t^{38} + t^{40} +
t^{42} + t^{44} + t^{46} + t^{48} + t^{50} + t^{52} + t^{56} + t^{60} + t^{64} + t^{68} + t^{70} + t^{72} + t^{74} + t^{76} + t^{78} + t^{80} + t^{82} + t^{84} + t^{88} + t^{92} +$$

$t^{95}+t^{96}+t^{100}+t^{102}+t^{104}+t^{106}+t^{108}+t^{110}+t^{111}+t^{112}+t^{114}+t^{116}+t^{120}+t^{124}+t^{125}+t^{128}+t^{132}+$
$t^{134}+t^{136}+t^{138}+t^{140}+t^{142}+t^{144}+t^{146}+t^{148}+t^{152}+t^{155}+t^{156}+t^{158}+t^{159}+t^{160}+t^{162}+t^{164}+t^{168}+$
$t^{171}+t^{172}+t^{174}+t^{175}+t^{176}+t^{178}+t^{180}+t^{182}+t^{184}+t^{185}+t^{186}+t^{187}+t^{188}+t^{189}+t^{190}+t^{192}+t^{194}+$
$t^{196}+t^{200}+t^{203}+t^{204}+t^{206}+t^{208}+t^{210}+t^{214}+t^{217}+t^{218}+t^{220}+t^{223}+t^{224}+t^{230}+t^{234}+t^{236}+t^{238}+$
$t^{239}+t^{240}+t^{242}+t^{244}+t^{245}+t^{248}+t^{252}+t^{253}+t^{256}+t^{260}+t^{262}+t^{264}+t^{266}+t^{268}+t^{270}+t^{272}+t^{274}+$
$t^{276}+t^{280}+t^{283}+t^{284}+t^{286}+t^{287}+t^{288}+t^{290}+t^{292}+t^{296}+t^{299}+t^{300}+t^{303}+t^{304}+t^{305}+t^{308}+t^{310}+$
$t^{312}+t^{313}+t^{314}+t^{315}+t^{316}+t^{317}+t^{318}+t^{320}+t^{322}+t^{324}+t^{328}+t^{331}+t^{332}+t^{335}+t^{336}+t^{337}+t^{342}+$
$t^{345}+t^{346}+t^{348}+t^{352}+t^{356}+t^{358}+t^{360}+t^{362}+t^{365}+t^{367}+t^{369}+t^{372}+t^{373}+t^{376}+t^{380}+t^{381}+t^{384}+$
$t^{388}+t^{390}+t^{392}+t^{394}+t^{396}+t^{399}+t^{400}+t^{401}+t^{404}+t^{408}+t^{411}+t^{412}+t^{414}+t^{416}+t^{418}+t^{427}+t^{429}+$
$t^{430}+t^{431}+t^{434}+t^{436}+t^{438}+t^{440}+t^{441}+t^{442}+t^{444}+t^{445}+t^{446}+t^{448}+t^{450}+t^{459}+t^{460}+t^{462}+t^{464}+$
$t^{466}+t^{468}+t^{470}+t^{472}+t^{473}+t^{474}+t^{475}+t^{476}+t^{479}+t^{480}+t^{485}+t^{486}+t^{490}+t^{492}+t^{494}+t^{495}+t^{496}+$
$t^{498}+t^{500}+t^{501}+t^{503}+t^{504}+t^{508}+t^{509}+t^{512}+t^{516}+t^{518}+t^{520}+t^{522}+t^{524}+t^{526}+t^{528}+t^{530}+t^{532}+$
$t^{536}+t^{539}+t^{540}+t^{542}+t^{543}+t^{544}+t^{546}+t^{548}+t^{552}+t^{555}+t^{556}+t^{559}+t^{561}+t^{566}+t^{568}+t^{569}+t^{570}+$
$t^{571}+t^{572}+t^{573}+t^{574}+t^{576}+t^{578}+t^{580}+t^{584}+t^{587}+t^{588}+t^{591}+t^{593}+t^{598}+t^{600}+t^{601}+t^{602}+t^{612}+$
$t^{614}+t^{616}+t^{618}+t^{621}+t^{623}+t^{624}+t^{625}+t^{629}+t^{632}+t^{635}+t^{636}+t^{637}+t^{638}+t^{640}+t^{642}+t^{644}+t^{648}+$
$t^{651}+t^{652}+t^{655}+t^{657}+t^{660}+t^{662}+t^{665}+t^{666}+t^{678}+t^{682}+t^{685}+t^{686}+t^{687}+t^{688}+t^{690}+t^{692}+t^{700}+$
$t^{701}+t^{702}+t^{706}+t^{715}+t^{718}+t^{720}+t^{722}+t^{725}+t^{726}+t^{728}+t^{729}+t^{730}+t^{731}+t^{732}+t^{734}+t^{735}+t^{738}+$
$t^{741}+t^{743}+t^{748}+t^{750}+t^{751}+t^{754}+t^{757}+t^{759}+t^{760}+t^{763}+t^{764}+t^{765}+t^{766}+t^{768}+t^{770}+t^{772}+t^{776}+$
$t^{779}+t^{780}+t^{782}+t^{786}+t^{788}+t^{790}+t^{793}+t^{794}+t^{799}+t^{804}+t^{806}+t^{808}+t^{810}+t^{812}+t^{815}+t^{816}+t^{817}+$
$t^{827}+t^{828}+t^{829}+t^{830}+t^{834}+t^{836}+t^{840}+t^{843}+t^{847}+t^{849}+t^{851}+t^{852}+t^{853}+t^{856}+t^{857}+t^{864}+t^{867}+$
$t^{868}+t^{871}+t^{872}+t^{877}+t^{878}+t^{879}+t^{882}+t^{884}+t^{885}+t^{888}+t^{892}+t^{893}+t^{896}+t^{900}+t^{902}+t^{904}+t^{906}+$
$t^{908}+t^{911}+t^{912}+t^{913}+t^{915}+t^{916}+t^{918}+t^{920}+t^{922}+t^{924}+t^{928}+t^{931}+t^{932}+t^{936}+t^{939}+t^{941}+t^{943}+$
$t^{945}+t^{950}+t^{953}+t^{954}+t^{955}+t^{956}+t^{957}+t^{958}+t^{960}+t^{962}+t^{965}+t^{971}+t^{972}+t^{974}+t^{976}+t^{978}+t^{980}+$
$t^{982}+t^{983}+t^{984}+t^{985}+t^{986}+t^{987}+t^{988}+t^{991}+t^{992}+t^{997}+t^{1001}+t^{1004}+t^{1006}+t^{1007}+t^{1008}+t^{1010}+$
$t^{1012}+t^{1013}+t^{1015}+t^{1016}+t^{1020}+t^{1021}+t^{1024}+t^{1028}+t^{1030}+t^{1032}+t^{1034}+t^{1036}+t^{1038}+t^{1040}+$
$t^{1042}+t^{1044}+t^{1048}+t^{1051}+t^{1052}+t^{1054}+t^{1055}+t^{1056}+t^{1058}+t^{1060}+t^{1064}+t^{1067}+t^{1068}+t^{1071}+$
$t^{1073}+t^{1078}+t^{1080}+t^{1081}+t^{1082}+t^{1083}+t^{1084}+t^{1085}+t^{1086}+t^{1088}+t^{1090}+t^{1092}+t^{1096}+t^{1099}+$
$t^{1100}+t^{1103}+t^{1105}+t^{1110}+t^{1112}+t^{1113}+t^{1114}+t^{1124}+t^{1126}+t^{1128}+t^{1130}+t^{1133}+t^{1135}+t^{1136}+t^{1137}+$
$t^{1139}+t^{1141}+t^{1144}+t^{1147}+t^{1148}+t^{1149}+t^{1150}+t^{1152}+t^{1154}+t^{1156}+t^{1160}+t^{1163}+t^{1164}+t^{1167}+t^{1169}$

and

$f_3 := x^2 + a_1 x + a_0 = x^2 + (t^{480} + w^2 t^{576} + ...)x + t^{96} + w^2 t^{192} + ...$
We point out that since the coefficients of f_3 consists of too many terms, we do not list them.

References

[1] Wieb Bosma, John Cannon, and Catherine Playoust. "The Magma algebra system. I. The user language." In: *J. Symbolic Comput.* 24.3-4 (1997). Computational algebra and number theory (London, 1993), pp. 235–265. ISSN: 0747-7171. DOI: 10.1006/jsco.1996.0125. URL: http://dx.doi.org/10.1006/jsco.1996.0125.

[2] Alin Bostan et al. "Fast computation of special resultants." In: *Journal of Symbolic Computation* 41.1 (2006), pp. 1–29. ISSN: 0747-7171. DOI: https://doi.org/10.1016/j.jsc.2005.07.001. URL: https://www.sciencedirect.com/science/article/pii/S0747717105001203.

[3] Alin Bostan et al. *Fast Computation with Two Algebraic Numbers.* Research Report RR-4579. INRIA, 2002. URL: https://inria.hal.science/inria-00072009.

[4] Jim Brown et al. "Classifying extensions of the field of formal Laurent series over $\mathbb{F}_p$." In: *Rocky Mountain Journal of Mathematics* 45 (Feb. 2015). DOI: 10.1216/RMJ-2015-45-1-115.

[5] Henri Cohen. *A Course in Computational Algebraic Number Theory.* Vol. 138. Graduate Texts in Mathematics. Berlin: Springer-Verlag, 1993. ISBN: 3–540–55640–0.

[6] Henri Cohen. *Advanced Topics in Computational Number Theory.* Vol. 193. Graduate Texts in Mathematics. Springer, New York, NY, 2000.

[7] David S. Dummit and Richard M. Foote. *Abdtract Algebra.* Third Edition. John Wiley and Sons, 2004.

[8] Roberto Dvornicich and Carlo Traverso. "Newton symmetric functions and the arithmetic of algebraically closed fields." In: *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes.* Ed. by Llorenç Huguet and Alain Poli. Berlin, Heidelberg: Springer Berlin Heidelberg, 1989, pp. 216–224. ISBN: 978-3-540-46150-0.

[9] Ivan B. Fesenko and Sergei V. Vostokov. *Local Fields and Their Extensions.* Second Edition. Vol. 121. American Mathematical Society, 2002.

[10] Claus Fieker and Jürgen Klüners. "Computation of Galois groups of rational polynomials." In: *LMS Journal of Computation and Mathematics* 17.1 (2014), pp. 141–158. URL: https://doi.org/10.1112/S1461157013000302.

[11] Patrick Fitzpatrick and Graham H. Norton. "The Berlekamp-Massey algorithm and linear recurring sequences over a factorial domain." In: *Applicable Algebra in Engineering, Communication and Computing* 6 (1995), pp. 309–323. DOI: 10.1007/BF01235722. URL: https://doi.org/10.1007/BF01235722.

[12] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra.* 3rd ed. Cambridge University Press, 2013. DOI: 10.1017/CBO9781139856065.

[13] Katharina Geißler and Jürgen Klüners. "Galois group computation for rational polynomials." In: *J. Symbolic Comput.* 30.6 (2000), pp. 653–674. ISSN: 0747-7171.

[14] Fernando Q. Gouvêa. "The p-adic Numbers." In: *p-adic Numbers: An Introduction.* Cham: Springer International Publishing, 2020, pp. 53–71. ISBN: 978-3-030-47295-5. DOI: 10.1007/978-3-030-47295-5_3. URL: https://doi.org/10.1007/978-3-030-47295-5_3.

[15] Christian Greve. "Galoisgruppen von Eisensteinpolynomen über p-adischen Körpern." Dissertation. Universität Paderborn, 2010.

[16] Christian Greve and Sebastian Pauli. “Ramification Polygons, Splitting Fields, and Galois groups of Eisenstein Polynomials.” In: *International Journal of Number Theory* 08 (Aug. 2012). DOI: 10.1142/S1793042112500832.

[17] Alfeen Hasmani et al. *Classifying extensions of the field of formal Laurent series over $\mathbb{F}_p$*. 2012. URL: http://www.math.clemson.edu/~kevja/REU/2012/REU%20writeup-ExtLocalFcFields.pdf.

[18] Helmut Hasse. *Number Theory*. Grundlehren der mathematischen Wissenschaften. Springer-Verlag, 1980. ISBN: 9783540082750. URL: https://books.google.de/books?id=xgPvAAAAMAAJ.

[19] Kenkichi Iwasawa. *Local Class Field Theory*. Oxford University Press, 1986.

[20] John W. Jones and David P. Roberts. “A database of local fields.” In: *Journal of Symbolic Computation* 41.1 (2006), pp. 80–97. ISSN: 0747-7171. DOI: https://doi.org/10.1016/j.jsc.2005.09.003. URL: http://www.sciencedirect.com/science/article/pii/S0747717105001276.

[21] Jürgen Klüners. “On computing subfields. A detailed description of the algorithm.” en. In: *Journal de théorie des nombres de Bordeaux* 10.2 (1998), pp. 243–271. URL: http://www.numdam.org/item/JTNB_1998__10_2_243_0.

[22] Jürgen Klüners. “Über die Berechnung von Teilkörpern algebraischer Zahlkörper.” Diplomarbeit. Technische Universität Berlin, 1995.

[23] Serge Lang. *Algebra*. 3rd ed. Springer, New York, NY, 2002. ISBN: 978-1-4613-0041-0. DOI: https://doi.org/10.1007/978-1-4613-0041-0.

[24] Adrien-Marie Legendre. *Théorie de Nombres*. Firmin Didot Frères, Paris, 1830.

[25] Ruediger Loos. “Computing in Algebraic Extensions.” In: *Computer Algebra: Symbolic and Algebraic Computation*. Ed. by Bruno Buchberger, George Edwin Collins, and Rüdiger Loos. Vienna: Springer Vienna, 1982, pp. 173–187. ISBN: 978-3-7091-3406-1. DOI: 10.1007/978-3-7091-3406-1_12. URL: https://doi.org/10.1007/978-3-7091-3406-1_12.

[26] Krasner Marc. “Sur la primitivité des corps p-adiques.” In: *Mathematica* 13 (1937), pp. 72–191.

[27] Thomas Mattman and John McKay. “Computation of Galois groups over function fields.” In: *Math. Comp.* 66.218 (1997), pp. 823–831. ISSN: 0025-5718.

[28] James S. Milne. *Algebraic Number Theory*. 2012. URL: www.jmilne.org/math/.

[29] Jonathan Milstead. “Computing Galois Groups of Eisenstein Polynomials Over P-adic Fields.” PhD thesis. University of North Carolina, Greensboro, Jan. 2017.

[30] Jonathan Milstead, Sebastian Pauli, and Brian Sinclair. “Constructing Splitting Fields of Polynomials over Local Fields.” In: *Collaborative Mathematics and Statistics Research*. Ed. by Jan Rychtář et al. Cham: Springer International Publishing, 2015, pp. 101–124. ISBN: 978-3-319-11125-4.

[31] Maurizio Monge. “A family of Eisenstein polynomials generating totally ramified extensions, identification of extensions and construction of class fields.” In: *International Journal of Number Theory* 10 (Sept. 2011). DOI: 10.1142/S1793042114500511.

[32] Patrick Morandi. *Field and Galois Theory*. Springer New York, 1996. DOI: https://doi.org/10.1007/978-1-4612-4040-2.

[33] Maruti Ram Murty. *Introduction to P-Adic Analytic Number Theory*. AMS/IP Studies in Advanced Mathematics Series. American Mathematical Soc., 2009. ISBN: 9780821888308. URL: https://books.google.de/books?id=nMYWZSYHITgC.

[34] Władysław Narkiewicz. *Elementary and Analytic Theory of Algebraic Numbers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. ISBN: 978-3-662-07001-7. DOI: 10.1007/978-3-662-07001-7_2. URL: https://doi.org/10.1007/978-3-662-07001-7_2.

[35] Jürgen Neukirch. *Algebraic Number Theory*. Springer, 1999.

[36] Öystein Ore. “Zur Theorie der Algebraischen Körper.” In: *Acta Mathematica* 44 (1923), pp. 219–314.

[37] Victor Y. Pan. "New Techniques for the Computation of Linear Recurrence Coefficients." In: *Finite Fields and Their Applications* 6 (2000), pp. 93–118.

[38] Sebastian Pauli. "Factoring Polynomials over Local Fields II." In: *Algorithmic Number Theory*. Ed. by Guillaume Hanrot, François Morain, and Emmanuel Thomé. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 301–315. ISBN: 978-3-642-14518-6.

[39] Michael Pohst and Hans Zassenhaus. *Algorithmic Algebraic Number Theory*. Cambridge University Press, 1989. ISBN: 0–521–330602.

[40] Steven Roman. *Field theory*. Second Edition. Vol. 158. Springer New York, NY, 2006.

[41] David Romano. "Galois groups of strongly Eisenstein polynomials." PhD thesis. University of California at Berkeley, U.S.A., May 2000.

[42] Joseph J. Rotman. *Advanced Modern Algebra*. Second Edition. Vol. 114. American Mathematical Society, 2010.

[43] Joseph J. Rotman. *An Introduction to the Theory of Groups*. Fourth Edition. Vol. 148. Springer New York, NY, 1995.

[44] John Scherk. "The Ramification Polygon for Curves over a Finite Field." In: *Canadian Mathematical Bulletin* 46 (Mar. 2003). DOI: 10.4153/CMB-2003-015-9.

[45] Arnold Schönhage. "Fast Parallel Computation of Characteristic Polynomials by Leverrier's Power Sum Method Adapted to Fields of Finite Characteristic." In: *International Colloquium on Automata, Languages and Programming*. 1993.

[46] Arnold Schönhage. "The fundamental theorem of algebra in terms of computational complexity - preliminary report." In: 1982.

[47] Brian Sinclair. "Algorithms for enumerating invariants and extensions of local fields." PhD thesis. University of North Carolina, Greensboro, Jan. 2015.

[48] Richard P. Stauduhar. "The determination of Galois groups." In: *Math. Comp.* 27 (1973), pp. 981–996. URL: https://doi.org/10.2307/2005536.

[49] Ian Stewart. *Galois Theory*. Third Edition. Charman and Hall/CRC Mathematics, 2003.

[50] Barry M. Trager. "Algebraic Factoring and Rational Function Integration." In: *Proceedings of the Third ACM Symposium on Symbolic and Algebraic Computation*. SYMSAC '76. New York, USA: ACM, 1976, pp. 219–226. URL: http://doi.acm.org/10.1145/800205.806338.

[51] Mark van Hoeij. "Factoring Polynomials and the Knapsack Problem." In: *Journal of Number Theory* 95.2 (2002), pp. 167–189. ISSN: 0022-314X. DOI: https://doi.org/10.1006/jnth.2001.2763. URL: https://www.sciencedirect.com/science/article/pii/S0022314X01927635.

List of Figures

List of Algorithms

Notation

A_n	Alternating group on n elements	
C_n	Cyclic group of order n	
D_n	Dihedral group of order 2n	
F_n	Frobenius group	
He	Heisenberg group	
K	Local function field of prime characteristic, i.e $k = K := \mathbb{F}_q((t))$, where $q = p^l$ and p is a prime number	
$K^\times$	$K^\times = K \setminus \{0\}$	
L/K	Field extension	
Q_8	Quaternion group	
SD	Semidihedral group	
S_n	Symmetric group on n elements	
$[L : K]$	Degree of the field extension L/K	
$\mathrm{Aut}(N)$	Automorphisms of the field N	
$\mathrm{Aut}(N/K)$	K-automorphisms of the field N	
$\mathrm{Char}(K)$	Characteristic of a field K	
$\mathrm{Gal}(f)$	Galois Group of a polynomial f	
$\mathrm{Spl}(f)$	Splitting field of a polynomial f	
$\min(\theta, k)$	Minimal polynomial of an element θ over k	
$\nu_{\max}$	Maximum valuation of the differences of the roots	93,
$\mathrm{prec}(F)$	Precision with respect of the indeterminate of the field F	
$e(L/K)$	Ramification index of the extension L/K	6,
v_K	(Exponential) Valuation of field K	3,
v_u	u-valuation	
M_α	$M_\alpha = \min\left(\{j \ : \ v_u(\alpha) \leq j \leq \nu_{\max} \mid p \nmid j, \ a_j \neq 0\} \cup \{\nu_{\max}\}\right)$	96,
$O_i = \mathrm{prec}(u_i)$	The u_i-precision	34,
$S_r(u)$	$S_r(u) := \sum_{i=1}^{r} d_i u^i \in \mathbb{F}_{\tilde{q}}[u]$	79,
S_κ	$S_\kappa := \sum_{i=1}^{n} \alpha_i^\kappa$ the κ-th higher trace (power sum) of the roots $\alpha_1, \ldots \alpha_n$, where $\kappa \in \mathbb{Z}_{\geq 0}$.	39,
$\mathbb{F}_q$	Finite field with q elements	
$\mathcal{NP}(f)$	Newton polygon of a polynomial f	11,
$\mathcal{O}_K$	Ring of integers of K	2,
$\mathcal{RP}(f)$	Ramification polygon of a polynomial f	17,
$\mathcal{U}_K$	The group of units of K	2,
$\mathcal{M}_K$	The unique maximal ideal of $\mathcal{O}_K$	2,
$m_{\mathcal{A}}$	The minimal polynomial $m_{\mathcal{A}} \in F[X]$ of a linear recurrent sequence $\mathcal{A} = (a_i)_{i=0}^{\infty} \in F^{\mathbb{N}}$	43,
$\mathfrak{F} = b_m$	It is the equation of $d_{m-\mathrm{val}+1}$ as defined in Proposition 6.1.15	82,
$\mathfrak{z}_{\mathfrak{p}}$	$\mathfrak{z}_{\mathfrak{p}} := w_{\mathfrak{p}} a_s \prod_{i=1}^{\bar{r}} d_{c_i}^{e_i}$. See Remark 6.1.12	82,
$\mathrm{Tr}_{k(\alpha)/k}$	The trace of $k(\alpha)/k$	47,
$\mathfrak{f}(L/K)$	Inertia Degree of the extension L/K	6,

$\operatorname{rev}(f)$	The reversal of a polynomial $f \in K[X]$, that is $\operatorname{rev}(f) = T^{\deg(f)} f(\frac{1}{X})$. If f is of degree at most n, then we write $\operatorname{rev}(n, f) = T^n f(\frac{1}{X})$ for its n-th reversal	40,
$\rho(g_{d_r})$	Number of distinct roots of g_{d_r}	
ρ_f	Ramification polynomial of a polynomial f	17,
$\sigma(u)$	$\sigma(u) = \sum_{i \geq 1} d_i u^i$ with $d_i \in \mathbb{F}_{\tilde{q}}$	79,
$\sim$	For $\gamma, \gamma' \in \bar{K}$ we write $\gamma \sim \gamma'$ if $v_K(\gamma - \gamma') > v_K(\gamma)$.	7,
τ_0	It is the leading partition of s. See Definition 6.1.28	86,
$\underline{A}_r(y)$	The residual polynomial of $f \in K[X]$ that corresponds to the segment S_r	20,
val	val $:= v_u(\alpha) = v_{u_\ell}(\alpha)$. Valuation of a root of a given polynomial	79,
g_{d_r}	The polynomial which corresponds to the equation of d_r for every $r \geq 1$	91,