Algorithms for Drinfeld Modules

by

Yossef Musleh

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2024

Examining Committee Membership

Author's Declaration

Statement of Contributions

Parts of the results presented in chapter 6 and 7 were originally published in [56] and [57], and are primarily my own work. The software presented in Appendix A was joint work with David Ayotte, Xavier Caruso, and Antoine Lèudiere and a demo was published in [4].

Abstract

Drinfeld modules play an important role in number theory over function fields, analogizing that of elliptic curves for the number field setting. The broad success in translating results over from number fields to function fields has motivated the study of Drinfeld modules, particularly in light of the enormous theoretical and practical weight of their elliptic curve counterparts. More recently, this has taken a more computational focus, with the potential for applications to polynomial factorization over finite fields. The main focus of this work will be to study the computation of the characteristic polynomial of the Frobenius endomorphism of a Drinfeld module. This problem has its roots in point counting problems over elliptic curves.

This thesis introduces several new algorithms for computing the characteristic polynomial of the Frobenius endomorphism. In particular, we give three new algorithms for computing the characteristic polynomial for Drinfeld modules of any rank, as well as two modifications to existing algorithms for the rank two case only. We also prove their correctness and analyze their bit complexity. In addition, we analyze algorithms for computing basis representations for the space of morphisms between Drinfeld modules derived from pre-existing approaches described in the literature.

Acknowledgements

I would like to thank the following individuals and groups for their contributions either directly to my academic work or to improving my quality of life. The order is somewhat chronological, and does not necessarily reflect the magnitude of their contribution. Some direct contributions to the thesis will be explicitly noted.

- My family; Mom, Dad, Omar, and Homoudy.
- My aunts, uncles, and cousins.
- My friends from Glenforest S.S.
- All the friends I've made at the University of Waterloo, particularly those I met through the Pure Math Club, UW HvZ, and the Avalon club.
- Éric, whose supervision made this thesis possible.
- Nora for sending me amusing content from the internet.
- The "Drinfeld crew" of David, Xavier, and Antoine for discussions and work on the implementation of Drinfeld modules.
- The members of the thesis committee for reviewing and approving this work.

Dedication

This thesis is dedicated to everyone who helped me get here.

Table of Contents

List of Tables

Chapter 1

Introduction

Since their initial introduction by Drinfeld [22], his eponymous modules, originally referred to as *elliptic modules*, have become important number theoretic tools. Originally motivated by efforts to prove special cases of the Langlands conjectures for global function fields, Drinfeld modules are now understood as just one component of a vast dictionary between the number field and function field setting. Spanning modular forms, L-functions, higher dimensional ablian varieties, and more, there has been considerable success in porting constructions and results over to the Drinfeld setting.

More recently, considerable amount of effort has gone in a computational direction, with much of it mirroring the existing machinery for elliptic curves. Some of the early work in this direction included Gekeler's work on the distribution of Frobenius traces of Drinfeld modules [34] who was motivated by potential parallels with the Sato-Tate conjecture for elliptic curves. The theory of zeta functions and their relation to counting rational points is a classical result that has motivated the design of algorithms for computing the number of points on an elliptic curve over a finite field. This is done by computing the characteristic polynomial of the *Frobenius endomorphism* of the curve, and has motivated a number of algorithms to carry out this operation; with the most famous being Schoof's algorithm for elliptic curves and its extensions. This provided one of the main entry points for my investigation into algorithms for computing the characteristic polynomial of the Frobenius endomorphism. Since then, the algorithmic study of Drinfeld modules has grown to include isogenies and isogeny graphs [13, 74], endomorphism rings [51, 29], and characteristic polynomials [34, 29, 59, 56]. Moreover, there have been attempts to translate cryptographic constructions paralleling the machinery over elliptic curves to the Drinfeld module setting.

While some readers may be more familiar with the general construction of Drinfeld modules, we will largely restrict our consideration to Drinfeld modules over $A = \mathbb{F}_q[x]$ for a finite field $\mathbb{F}_q$ of cardinality q. We will exclusively work over finite Drinfeld modules; that is, fix a finite extension $\mathbb{L}$ of $\mathbb{F}_q$ with $n = [\mathbb{L} : \mathbb{F}_q]$ and a choice of *characteristic map* $\gamma : A \to \mathbb{L}$. The map γ necessarily has a non-trivial kernel generated by a monic irreducible $\mathfrak{p} \in A$ referred to as the *characteristic* or *A-characteristic*. Now let τ be a non-commutative indeterminate such that, for elements $a \in \mathbb{L}$, $\tau a = a^q\tau$, and let $\mathbb{L}\{\tau\}$ denote the ring of polynomials in τ with coefficients in $\mathbb{L}$. Then a Drinfeld module of rank r can be realized concretely as an $\mathbb{F}_q$ algebra homomorphism $\phi : A \to \mathbb{L}\{\tau\}$ such that $\phi(x) = \sum_{i=1}^{r} \Delta_i \tau^i + \gamma(x)$ with $\Delta_r \neq 0$. Morphisms between Drinfeld modules with domain and codomain ϕ, ψ respectively can be realized concretely as elements $u \in \mathbb{L}\{\tau\}$ such that $u\phi_x = \psi_x u$. To each endomorphism of a Drinfeld module, one can associate a characteristic polynomial and minimal polynomial, denoted $\mathrm{CharPoly}(u)$ and $\mathrm{MinPoly}(u)$, arising from its induced action on the so-called *Tate module*, the details of which are discussed in more detail in chapter 4.

When stating our complexity results, we will let $\omega < 2.372$ denote the exponent of matrix multiplication; thats is, a real number such that two square matrices of size d over a ring can be multiplied in $O(d^\omega)$ ring operations. Similarly, let $\omega_2 < 3.2516$ denote the exponent of $d \times d$ by $d \times d^2$ matrix multiplication, and λ the exponent of taking the characteristic polynomial of a matrix. These objects are discussed in more detail in sections 2.2.1 and 2.2.2. We will also let $\mathrm{SM}(d, n, q)$ denote the complexity of multiplying two skew polynomials in $\mathbb{L}\{\tau\}$ of degree at most d; further details on the complexiy of this operation are discussed in section 2.4.

The main goal of this thesis is to present new algorithms for computing the characteristic polynomial of endomorphisms of a finite Drinfeld module, particularly the *Frobenius endomorphism* corresponding to the skew polynomial τ^n which is sometimes referred to as *the* characteristic polynomial of a Drinfeld module. Chapter 2 will cover the prerequisite algorithmic theory concerning skew polynomials on which our main algorithms will depend, and in particular will establish much of the basic complexity concepts and notation that we will use in our analysis of algorithms. Chapter 3 will review some of the basic theory of elliptic curves, and describe the computation of the characteristic polynomial in this setting. Chapter 4 will provide the required background theory on Drinfeld modules and will include an analysis of previous algorithms for computing the characteristic polynomial of endomorphisms of a Drinfeld module. Chapter 5 will introduce the notion of crystalline cohomology of Drinfeld modules.

Chapter 6 will introduce several new results; in particular we will present several new approaches to compute the characteristic polynomial of the Frobenius endomorphism in any rank, as well as modifications to existing algorithms for the rank 2 case. The following theorem extends the previous "Schoof-like" algorithm which was originally presented in the "large base-field" case only in [55].

Theorem (6.1.1). *There exists a deterministic algorithm to compute the characteristic polynomial of the Frobenius endomorphism of a rank 2 Drinfeld module ϕ with a bit complexity of $(n^2 \log q + n \log^2 q)^{1+o(1)}$.*

This extends the approach of [55, Thm. 15], which previously required $q > \frac{n}{2}$. In addition, we will present a modification of a pre-existing algorithm given in [59] which required that the minimal polynomial of ϕ_x as a linear operator on $\mathbb{L}/\mathbb{F}_q$ have degree n.

Theorem (6.2.1). *Let ϕ_x be a rank 2 Drinfeld module over $(\mathbb{L}, \gamma)$. There is a Monte Carlo algorithm to compute the characteristic polynomial of the Frobenius endomorphism of ϕ if $\deg \operatorname{MinPoly}(\phi_x) = n$ with a bit complexity of $(n^{1.885} \log q + n \log^2 q)^{1+o(1)}$.*

We will present generalizations of two algorithms, previously only applicable to the rank two case only, to Drinfeld modules of arbitrary rank. The first result is a generalization of the "Schoof-like" algorithm seen in [55].

Theorem (6.1.2). *Let ϕ be a rank r Drinfeld module over $\mathbb{L}$ such that $\operatorname{CharPoly}(\tau^n) = \operatorname{MinPoly}(\tau^n)$. There exists an algorithm such that if $\gcd(n, r) = 1$ or $\gcd(n, r-1) = 1$, and $q > N$, then the algorithm computes the characteristic polynomial of the Frobenius endomorphism of ϕ if the following conditions hold.*

1. *ϕ_x lies outside of a hypersurface in $\mathbb{F}_q^{n(r+1)}$ defined by a polynomial $\mathfrak{f}$ of degree at most $(n-1)(r-1)$.*

2. *A randomly chosen vector $E = (e_1, \dots, e_n) \in \mathbb{F}_q^N$ lies outside of a hypersurface in $\mathbb{F}_q^N$ of degree at most $n^2 r + \frac{n^2}{2}$.*

3. *A randomly chosen projection $\ell : \mathbb{L} \to \mathbb{F}_q$ lies outside of a hypersurface in $\mathbb{F}_q^n$ of degree at most $n(r-1)$.*

Moreover, this algorithm runs with a bit complexity of $(r^{\omega_2/2+1} n^2 \log q + n \log^2 q)^{1+o(1)}$.

This improves the complexity of the characteristic polynomial to quadratic order in the degree of the field extension $n = [\mathbb{L} : \mathbb{F}_q]$, which was previously cubic using either the "direct approach" or that of [29]. The second result generalizes an algorithm based on structured linear systems again seen in [55].

Theorem (6.3.1)**.** *Let ϕ be a Drinfeld module of rank r over $\mathbb{L}$. There exists a Monte Carlo algorithm to compute the characteristic polynomial of the Frobenius endomorphism when $\gcd(n, r) = 1$ with a bit complexity of $(n^3 r \log q)^{1+o(1)}$ and which returns the correct result with probability at least $\left(1 - \frac{n^2 r^2}{q}\right)$.*

The final major result on this topic is a series of deterministic algorithms for computing the characteristic polynomial of any endomorphism of a Drinfeld module of any rank based on the cohomological constructions due to Anglès and Gekeler [2]. These algorithms will then be specialized to the case of the Frobenius endomorphism, as well as the "prime field case" where $\mathbb{F}_\mathfrak{p} = \mathbb{L}$. The results are summarized in the following theorem.

Theorem (6.4.1)**.** *Let ϕ be a Drinfeld $\mathbb{F}_q[x]$-module over $(\mathbb{L}, \gamma)$, and let u be any endomorphism of ϕ of τ-degree d. Then there are deterministic algorithms to compute* $\mathrm{CharPoly}(u)$ *with the following complexities*

1. $\left(\frac{\min(dr^2, (d+r)r^{\omega-1})}{m}(d+m)n \log q + r^\lambda n(d+m)/m \log q + n \log^2 q\right)^{1+o(1)}$

2. $(r\mathrm{SM}(d+r, n, q) + r^\lambda n(d+m)/m \log q + n \log^2 q)^{1+o(1)}$ *which is either*

- *$(r(d+r)^{\omega_2/2} n \log q + r^{\lambda} n(d+m)/m \log q + n \log^2 q)^{1+o(1)}$ if $d < n$*
- *or $(r(d+r)n^{\omega-1} \log q + r^{\lambda} n(d+m)/m \log q + n \log^2 q)^{1+o(1)}$ otherwise*

3. *$((r^{\lambda}/m + r^{\omega}/\sqrt{m})n^2 \log q + n \log^2 q)^{1+o(1)}$ if $u = \tau^n$*

4. *$(r^{\omega} n^{1.5} \log q + n \log^2 q)^{1+o(1)}$ if $u = \tau^n$ and $\mathbb{L} = \mathbb{F}_{\mathfrak{p}}$.*

In the case of the Frobenius endomorphism, these algorithms again improve on the previously known optimal complexities for the characteristic polynomial computation, but have the key advantages of being deterministic, being generally applicable in any rank, and achieving a complexity on the order of $n^{1.5}$ when r is fixed in the prime field case. This last point is notable, as previously an algorithm achieving this bound was only available for the rank 2 case [34, §3.4]. Moreover, when r is fixed their worst case complexities are $O(n^2)$ which is a significant improvement over both the direct approach and the algorithm of [29]. For general endomorphisms, when n and r are fixed, algorithm 2 achieves a complexity of $d \log d$ in the degree of the endomorphism.

Chapter 7 will cover some additional results. We present a new algorithm for multi-point evaluation of skew polynomials; this result is contained in the following theorem.

Theorem (7.1.1)**.** *Given a skew polynomial $s \in \mathbb{L}\{\tau\}$ of degree d and t evaluation points $a_1, \ldots, a_t \in \mathbb{L}$, there exists an algorithm for computing the evaluations $s(a_1), \ldots, s(a_t)$ in time*

- $(dt^{\omega-2} n \log q)^{1+o(1)}$ if $t \leq \sqrt{d}$.
- $(d^{(\omega-1)/2} tn \log q)^{1+o(1)}$ otherwise.

In addition we will discuss the complexity of the algorithm of Wesolowski [74] for computing a basis of morphisms $\operatorname{Hom}(\phi, \psi)_d$ of Drinfeld modules ϕ, ψ of degree at most d, as well as extensions of the algorithm for computing a basis of the free module of morphisms over the coefficient rings $\mathbb{F}_q[\tau^n]$ and $\mathbb{F}_q[x]$. The main contribution here is a complete bit complexity analysis of the following algorithms for computing basis representations of the module of morphisms of Drinfeld modules ϕ, ψ over various coefficient rings.

Theorem (7.3.1)**.** *There is an algorithm for computing a basis of*

- $\mathrm{Hom}(\phi, \psi)_d$ *over* $\mathbb{F}_q$ *using* $(d^\omega n^\omega \log q + dn^3 r \log q + n \log^2 q)^{1+o(1)}$ *bit operations.*
- $\mathrm{Hom}(\phi, \psi)$ *over* $\mathbb{F}_q[\tau^n]$ *using* $(n^4 r \log q + n \log^2 q)^{1+o(1)}$ *bit operations.*
- $\mathrm{Hom}(\phi, \psi)$ *over* $\mathbb{F}_q[x]$ *using* $(n^\omega r^{\omega+1} \log q + n^3 r^3 \log q + n \log^2 q)^{1+o(1)}$ *bit operations.*

Appendix A will also contain a brief overview of an implementation of Drinfeld modules introduced in [54] which is now part of the SageMath implementation of Drinfeld modules. This will include an overview of some of the key features available in the current public distribution of SageMath, as well as some features still under development.

Chapter 2

Background

2.1 Elementary Mathematical Notation and Definitions

We will assume that the reader is familiar with basic algebraic structures, particularly groups, rings, and fields. We will fix here some of the most common mathematical notation used throughout this work. We will use R to denote an arbitrary commutative ring. Let $R[x]$ denote the set of polynomials with coefficients lying in R, and let $R[x]_d$ denote the set of polynomials of degree at most d. Furthermore, let $M_{s\times d}(R)$ and $M_d(R)$ denote the sets of $s \times d$ and $d \times d$ matrices respectively with coefficients in R. If $\iota : R \to R$ is a ring endomorphism, this induces endomorphisms on $R[x]$, $M_{s\times d}(R)$ given by coefficient-wise action of ι. Given an element $f \in R[x]$ or $f \in M_{s\times d}(R)$ we will denote the action of ι in this way by f^{ι}.

In this work, $\mathbb{F}_q$ will denote a finite field of size q a prime power, and $\mathbb{L}$ will be used to denote a field extension of $\mathbb{F}_q$ of order $n = [\mathbb{L} : \mathbb{F}_q]$. We write $\mathbb{L}/\mathbb{F}_q$ to denote the $\mathbb{F}_q$-vector space structure on $\mathbb{L}$.

We will let $\mathrm{End}(R)$, $\mathrm{Hom}(R, R')$, $\mathrm{Aut}(R)$ denote the set of ring homomorphisms, endomorphisms, and automorphisms respectively, with identical notation when R is understood as a field. When R is an algebra over $\mathbb{F}_q$, we will write $\mathrm{End}_{\mathbb{F}_q}(R)$ for the $\mathbb{F}_q$-endomorphisms which fix $\mathbb{F}_q$, with similar notation $\mathrm{Hom}_{\mathbb{F}_q}(R, R')$, $\mathrm{Aut}_{\mathbb{F}_q}(R)$ for homomorphisms and automorphisms. If $\sigma \in \mathrm{Aut}(R)$, then the *order* of σ, denoted $\mathrm{ord}(\sigma)$ is the smallest positive integer g such that $\sigma^g = \mathrm{id}$.

Recall that $\mathbb{L}$ is a vector space of dimension n over $\mathbb{F}_q$, and any $s \in \mathbb{L}$ induces an $\mathbb{F}_q$-linear operator on $\mathbb{L}$ given by multiplication with s. The *norm* and *trace* of s, denoted $N_{\mathbb{L}/\mathbb{F}_q}(s)$ and $\mathrm{Tr}_{\mathbb{L}/\mathbb{F}_q}(s)$, are the determinant and trace of this linear operator respectively.

2.2 Elementary Algorithmics

As the primary goal of this thesis is to construct and analyze algorithms, we will define the complexity model used to state the runtimes of operations. We will make use of the standard asymptotic notation conventions, including $O(f), \tilde{O}(f)$, and $f^{1+o(1)}$ throughout this work, which we shall define here.

Definition 2.2.1. *Let $f : \mathbb{R}^+ \to \mathbb{R}^+, g : \mathbb{R}^+ \to \mathbb{R}^+$ be positive real valued functions. We say that*

- *$f \in O(g)$ if there exists constants $C, x_0 > 0$ such that $f(x) \leq Cg(x)$ for all $x > x_0$.*
- *$f \in \tilde{O}(g)$ if there exists a $k > 0$ such that $f \in O(g \log^k g)$.*
- *$f \in g^{1+o(1)}$ if there exists a function $h : \mathbb{R}^+ \to \mathbb{R}^+$ such that $\lim_{x\to\infty} h(x) = 0$ and $f \in O(g^{1+h})$.*

The two complexity models we will consider for this work are:

1. the algebraic model for rings and fields, in which the elementary operations of addition, multiplication, and division can be done at unit cost [11]
2. using a bit complexity model, which counts bit operations using a standard random access machine [61].

The algebraic operation model is the norm for most research in symbolic and algebraic computation, and we will make use of it at certain points. However, this work will primarily aim to give the runtimes of core algorithms in the bit complexity model. The reason for this, as will be discussed shortly, is the Kedlaya-Umans algorithm for modular composition, which is impossible to account for in the algebraic model and which plays an important role in the complexity of operations on skew polynomials. Using the bit complexity model, we will assume that the elementary field operations in $\mathbb{F}_q$ can be performed in $\tilde{O}(\log q)$ or $(\log q)^{1+o(1)}$ bit operations [30]. Consequently, algorithms with an algebraic cost of $O(t)$ operations in $\mathbb{F}_q$ can be assigned a bit complexity of $\tilde{O}(t \log q)$ or $(t \log q)^{1+o(1)}$.

2.2.1 Multiplication of Polynomials and Matrices

For rings which admit an FFT-like algorithm, we may take the complexity of multiplying two polynomials in $R[x]$ of degree at most d to be $O(d \log d)$ algebraic operations. For arbitrary rings, this can be done in $O(d \log d \log \log d)$ [12]. In particular, we will let $\mathrm{Mul}(n, q)$ denote the complexity of multiplying two finite field elements in a degree n extension of $\mathbb{F}_q$; based on the preceding statement, we may set $\mathrm{Mul}(n, q) = (n \log q)^{1+o(1)}$.

We will let $\omega \in \mathbb{R}$ denote a real number such that two elements of $M_d(R)$ can be multiplied in $O(d^\omega)$ algebraic operations. As of the writing of this work, the smallest bound given in the literature is $\omega \approx 2.372$ [23]. We will similarly let ω_2 denote the exponent such that an element of $M_d(R)$ can be multiplied by an element of $M_{d\times d^2}(R)$ in $O(d^{\omega_2})$ R-operations. A known bound is $\omega_2 < 3.2516$ [28] and in general $\omega_2 \leq \omega + 1$. For general multiplication of matrices of size $(d, t) \times (t, u)$, we can divide the factors into square blocks of size $\min(d, t, u)$ while padding as necessary. Using naive rectangular multiplication on the block decomposition yields an algorithm with a cost of $O(dtu \min(d, t, u)^{\omega-3})$ field operations.

2.2.2 Characteristic Polynomials

We will let λ denote the exponent such that the complexity of computing the characteristic polynomial of a $d \times d$ matrix with coefficients in R is $O(d^\lambda)$ ring operations. When R is a field, the characteristic polynomial of any matrix can be computed at a cost of $\tilde{O}(d^\omega)$ [11]. For more general rings, we will rely on the algorithm of Kaltofen and Villard [47] with a complexity corresponding to $\lambda \approx 2.7$.

2.2.3 Representations of Finite Field Elements

A salient detail of algorithms over finite fields is the choice of representation of field elements. For a field extension $\mathbb{F}_q \subset \mathbb{L}$ with $[\mathbb{L} : \mathbb{F}_q] = n$, which we may denote $\mathbb{F}_q \underline{\ \ n\ \ } \mathbb{L}$, this is typically done by first fixing a representation of $\mathbb{F}_q$ and representing elements of $\mathbb{L}$ as polynomials in $\mathbb{F}_q[x]/(\ell(x))$ for some choice of irreducible $\ell(x) \in \mathbb{F}_q[x]$ of degree n. The technicalities of this choice become clearer when one works over a *tower* of fields $\mathbb{F}_q \underline{\ \ m\ \ } \mathbb{F}_\mathfrak{p} \underline{\ n/m\ } \mathbb{L}$ with $[\mathbb{L} : \mathbb{F}_q] = n$, $[\mathbb{F}_\mathfrak{p}, \mathbb{F}_q] = m$ and $\mathbb{F}_\mathfrak{p} = \mathbb{F}_q[x]/(\mathfrak{p}(x))$ for an irreducible $\mathfrak{p}(x) \in \mathbb{F}_q[x]$. Then $\mathbb{L}$ may be represented in either univariate form as $\mathbb{L} = \mathbb{F}_q[t]/(\ell(t))$ with $\ell(t) \in \mathbb{F}_q[t]$ or bivariate form $\mathbb{L} = \mathbb{F}_q[x,t]/(\mathfrak{p}(x), g(x,t))$. The univariate representation is the default for most algorithms, and when necessary we may convert between the bivariate and univariate representations at the cost of a single modular composition; an algorithm which is discussed in section §2.3.

2.2.4 Companion Matrices

Given a sequence $(x_i)_{i\geq 0}$ defined by a length d recurrence with $a = (a_0, \ldots, a_{d-1})$ such that $x_j = \sum_{i=0}^{d-1} a_i x_{j-d+i}$ for $j \geq d$, we define the $d \times d$ *companion matrix* $\mathcal{C}_a$ to be

$$\begin{bmatrix} a_{d-1} & a_{d-2} & \ldots & a_1 & a_0 \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{bmatrix}.$$

Then we have that:

$$\begin{bmatrix} x_{j+1} \\ x_j \\ \vdots \\ x_{j-d} \end{bmatrix} = \mathcal{C}_a \begin{bmatrix} x_j \\ x_{j-1} \\ \vdots \\ x_{j-d+1} \end{bmatrix}$$

If we have a monic $a \in R[x]_d$ with $a = \sum_{i=0}^{d} a_i x^i$, the companion matrix of the polynomial has the form:

$$\mathcal{C}_a = \begin{bmatrix} -\frac{a_{d-1}}{a_d} & -\frac{a_{d-2}}{a_d} & \dots & -\frac{a_1}{a_d} & -\frac{a_0}{a_d} \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}.$$

2.3 Modular Composition

After multiplication and Euclidean division, perhaps the most natural problem to consider is that of *modular composition*

Operation 1. *Let R be a ring, and let $f, g, h \in R[x]$ be polynomials of degree at most d with h monic. The modular composition operation computes the polynomial*

$$f(g) \bmod h$$

For nearly 30 years, the best known algorithm for modular composition was based on the algorithm of Brent and Kung [10] who gave an algebraic complexity of $O(d^{(\omega+1)/2})$. This was later improved slightly by Huang and Pan in [41] to $O(d^{\omega_2})$.

This question was addressed once more in Kedlaya and Umans' paper [50], in which they gave algorithms for key operations including finding normal bases, polynomial factorization over finite fields, and modular composition, based on a novel and quasi-optimal algorithm for multivariate multipoint evaluation. The algorithm may be required to change the ring over which algebraic operations are performed, therefore making it impossible to give a complete accounting of the computational cost using an algebraic complexity model. Consequently, the runtimes for the algorithms are given purely in terms of bit complexity. Moreover, the algorithm has proven difficult to realize in the form of an efficient practical implementation, which will have implications for the timings of our implementations.

For the duration of this thesis, we will take the runtime of modular composition to be $(d \log q)^{1+o(1)}$ for polynomials over a finite field $\mathbb{F}_q$, based on a refinement of the Kedlaya-Umans algorithm [40].

2.3.1 Computing Frobenius Images

Given a field $\mathbb{F}$ of characteristic p, the Frobenius operator $\sigma_q : \mathbb{F} \to \mathbb{F}$ of order $q = p^k$ sends a to a^q. This operation acts linearly on fields of characteristic p, and if $\mathbb{F}_q \subset \mathbb{F}$ then σ_q is an $\mathbb{F}_q$-vector space isomorphism. For a fixed choice of Frobenius operator σ_q, for any $a \in \mathbb{F}$ let $a^{[i]} = \sigma_q^i(a) = a^{q^i}$. For $m \in M_{d\times d'}(\mathbb{F})$ $m^{[i]}$ denotes entry-wise application of σ_q^i, and for $f \in \mathbb{F}[x]$, $f^{[i]}$ denotes coefficient-wise application. In [32], von zur Gathen and Shoup gave an explicit approach reducing the computation of Frobenius maps on finite fields to modular composition via the so-called *polynomial representation of the Frobenius*.

Consider the representation of $\mathbb{F}$ presented as the set of modular polynomials $\mathbb{F}_q[x]/h(x)$ with $h(x)$ irreducible of degree n. The *Frobenius element* is $x^q \bmod h(x)$, and given any other element $f(x) \bmod h(x)$, we observe that $f(x^q) = f(x)^q \bmod h(x)$. Therefore, it suffices to compute $x^q \bmod h(x)$ once and then perform a single modular composition to compute the q-order Frobenius of any element of $\mathbb{F}$. Moreover, the q^i-order Frobenius element $x^{q^i} \bmod h$ can be obtained in a similar manner by successive modular compositions of $x^{q^{i-1}} \bmod h(x)$ with $x^q \bmod h(x)$. Thus, after accounting for an initial cost of $(n \log(q)^2)^{1+o(1)}$ bit operations to compute $x^q \bmod h(x)$, all further q-order Frobenius powers can be computed at the cost of modular composition. Therefore q^i-order Frobenius powers can be reduced to modular composition after a pre-computation step with total bit complexity $(n \log(q)^2 + n^2 \log q)^{1+o(1)}$.

2.3.2 Normal Bases

Recalling $\mathbb{L}/\mathbb{F}_q$ a field extension of order n, and let $\sigma : \mathbb{L} \to \mathbb{L}$ be the q-order Frobenius endomorphism on $\mathbb{L}$ $\sigma(a) = a^q$.

Definition 2.3.1. *A basis for $\mathbb{L}$ over $\mathbb{F}_q$ is said to be **normal** if it has the form $\{b, \sigma(b), \ldots, \sigma^{n-1}(b)\}$ for some $b \in \mathbb{L}$.*

Operation 2. *Given a degree n extension $\mathbb{L}/\mathbb{F}_q$ and σ is the q-order Frobenius, find a normal basis.*

We denote the complexity of computing a normal basis for σ of $\mathbb{L}/\mathbb{F}_q$ by $\mathrm{NB}(n, q, \sigma)$; we drop the dependence on σ when $\sigma(\cdot) = \cdot^q$. von zur Gathen and Giesbrecht [31] gave a Las Vegas algorithm for computing normal bases with expected $O(\mathrm{Mul}(n^2, q)(\log n)^2 + \mathrm{Mul}(n, q)\log q)$ $\mathbb{F}_q$-operations. Kaltofen and Shoup in [46] gave the first subquadratic algorithm for normal basis construction and conversion to and from power bases. This approach utilised a baby-step/giant-step method based on their improved algorithms automorphism projections. For a normal basis with respect to the Frobenius automorphism, and leveraging the Kedlaya-Umans algorithm for modular composition, the bit complexity of constructing and manipulating normal bases is bounded by

$$\mathrm{NB}(n, q) = n^{\omega_2/2} \log^{1+o(1)} q + n^{1+o(1)} \log^{2+o(1)} q$$

2.4 Skew Polynomials

As much of the mathematical theory of Drinfeld modules can be described quite concretely in terms of *skew polynomials* in the Frobenius operator, many algorithms of interest on Drinfeld modules rely on fundamental algorithms on skew polynomials. To that end, it will be necessary to describe and analyze the complexity of the key operations we will be calling upon in this work. Let $\sigma : R \to R$ be a ring endomorphism. A *σ-derivation* is a map $\delta : R \to R$ such that, for all $a, b \in R$

- $\delta(a+b) = \delta(a) + \delta(b)$
- $\delta(ab) = \sigma(a)\delta(b) + \delta(a)b.$

We can now define the ring of skew polynomials.

Definition 2.4.1. *Given a ring R, an endomorphism $\sigma : R \to R$ and a σ-derivation $\delta : R \to R$, let τ be an indeterminate such that $\tau a = \sigma(a)\tau + \delta(a)$ for all $a \in R$. We then define the ring of* ***Ore*** *or* ***skew polynomials*** *to be the set*

$$R\{\tau; \sigma, \delta\} = \left\{ \sum_{i=0}^{d} a_i \tau^i | a_i \in R, d \in \mathbb{Z}^+ \right\}$$

When $\delta = 0$, we will write $R\{\tau; \sigma\}$ instead of $R\{\tau; \sigma, 0\}$. When the choice of σ, δ is clear from context, we will suppress them from the notation and write the ring of skew polynomials as $R\{\tau\}$. Furthermore, there is a canonical identification $\xi : R\{\tau\} \to \mathrm{End}(R)$ given by

$$\sum_i a_i \tau^i \mapsto \sum_i a_i \sigma^i \in \mathrm{End}(R),$$

where $\sigma^0 = \mathrm{id}$. Then ξ can be seen as an $\mathbb{F}_q$-algebra homomorphism where ordinary multiplication of skew polynomials in $R\{\tau\}$ corresponds to composition of operators in $\mathrm{End}_{\mathbb{F}_q}(R)$. For $a \in \mathbb{L}$ we will simply write $\rho(a)$ to mean $\xi(\rho)(a)$.

We will denote the cost of computing of a single application of σ to any element by SK(..) in the relevant parameters, which for our finite field setting will be n, q. The complexities for skew polynomial operations in this chapter will be given in terms of $\mathrm{SK}(n, q)$ to keep the analysis general. However, for all algorithms concerning Drinfeld modules, including chapter 4 and onwards, σ will be the q-order Frobenius endomorphism $\sigma(x) = x^q$, in which case $\mathrm{SK}(n, q) = (n \log q)^{1+o(1)}$.

For an element $S \in R\{\tau\}$ whose leading coefficient is invertible in R, we let $\deg S$ denote its degree in τ, and $R\{\tau\}_d = \{S \in R\{\tau\} \mid \deg S \le d\}$. Given $T, S \in R\{\tau\}$, we shall define the *right quotient* T/S and *right remainder* $T \bmod S$, to be the unique values $Q, U \in R\{\tau\}$ respectively such that $T = QS + U$ with $\deg U < \deg S$. For fixed $S \in R\{\tau\}$ the right quotient $R\{\tau\}/S$ is a left $R\{\tau\}$-module in the usual manner. It will also be convenient to view elements of $R\{\tau\}/S$ as an R-module, and in our typical setting of $R = \mathbb{F}_q$ or $R = \mathbb{L}$ we have a natural vector space structure of dimension d. For any $s \in \mathbb{L}\{\tau\}$, we will use $\hat{s}$ to denote the corresponding representation as a column vector of size d.

For the remainder of this work, we will work exclusively with $\mathbb{L}\{\tau\}$. We will set σ to be the q-order Frobenius operator $\sigma_q(a) = a^q$ and set $\delta = 0$. In this setting, we have a further interpretation of skew polynomials as the so called *linearized polynomials*, under the identification

$$\sum_i a_i \tau^i \mapsto \sum_i a_i x^{q^i} \in \mathbb{F}_q[x].$$

2.4.1 Multiplication

Operation 3. *Let* $a = \sum_{i=0}^{s} a_i \tau^i, b = \sum_{i=0}^{s} b_i \tau^i \in \mathbb{L}\{\tau\}$. *The skew multiplication operation* $a \cdot b$ *returns the skew polynomial* $\sum_{0 \le i,j \le s} a_i \sigma^i(b_j) \tau^{i+j} = \sum_{d=0}^{2s} \sum_{i=0}^{\min(d,s)} a_i \sigma^i(b_{d-i}) \tau^d$.

We will denote the complexity of skew multiplication of skew degree d polynomials over a field extension $\mathbb{L}/\mathbb{F}_q$ such that $[\mathbb{L} : \mathbb{F}_q] = n$ by $\mathrm{SM}(d, n, q)$. We will now proceed to state and analyze two algorithms, due to Puchinger & Wachter-Zeh and Caruso & Le Borgne to give upper estimates on $\mathrm{SM}(d, n, q)$.

The Puchinger & Wachter-Zeh Algorithm

We begin with an analysis of the algorithm given in [64, Th.7]. The algorithm and proof of correctness are as given in the text; the full proof is given here as part of a refinement of the complexity estimates.

Theorem 2.4.2. *Two skew polynomials* $a, b \in \mathbb{L}\{\tau\}$ *can be multiplied with a bit complexity of* $d^{\omega_2/2}\mathrm{Mul}(n, q) + d^{3/2}\mathrm{SK}(n, q) = (d^{\omega_2/2} n \log q)^{1+o(1)}$.

Proof. Write our inputs as

$$a = \sum_{i=0}^{s} a_i \tau^i$$
$$b = \sum_{i=0}^{s} b_i \tau^i.$$

Let $d^* = \lceil \sqrt{d+1} \rceil$. We can split a into a sum of d^* terms of the form $a^{(i)} = \sum_{j=0}^{d^*-1} a_{id^*+j}\tau^{id^*+j}$ such that $a = \sum_{i=0}^{d^*-1} a^{(i)}$. This in turn splits the product c into d^* terms $c = \sum_{i=0}^{d^*-1} c^{(i)}$ with $c^{(i)} = a^{(i)}b$. We can multiply through the expression for $c^{(i)}$:

$$\begin{aligned} c^{(i)} &= \sum_{j=0}^{d^*-1} a_{id^*+j}\tau^{id^*+j}\left(\sum_{k=0}^{d} b_k \tau^k\right) \\ &= \sum_{k=0}^{d+d^*-1}\left(\sum_{j=0}^{k} a_{id^*+j}\sigma^{id^*+j}(b_{k-j})\tau^{id^*+k}\right) \\ &= \sum_{k=0}^{d+d^*-1} c_k^{(i)}\tau^{id^*+k} \end{aligned}$$

We now define matrices A, B, C such that:

$$\begin{aligned} A_{i,j} &= \sigma^{-id^*}(a_{id^*+j}) \\ B_{i,k} &= \sigma^{i}(b_{k-i}) \\ C_{i,k} &= \sigma^{-id^*}(c_k^{(i)}) \end{aligned}$$

Where $0 \le i, j \le d^* - 1$, $0 \le k \le d + d^* - 1$. Observe that $C = A \cdot B$; the coefficients $c_k^{(i)}$ can be recovered through $O(d^{3/2})$ applications of automorphisms and c can be recovered by summing the $c_k^{(i)}$.

Assuming the automorphism σ admits a pre-computation step similar to the polynomial representation, at the cost of a single application, representations of $\sigma^{\pm id^*}$, σ^j for $i < d^*$, $j < d + d^*$ can be computed in $O(d\mathrm{SK}(n, q))$. Taking advantage of fast rectangular matrix multiplication, we can compute the matrix product in $O(d^{\omega_2/2}\mathsf{M}(n,q))$. Applying automorphisms to compute the entries of A, B, as well as to recover the terms $c_k^{(i)}$ from C costs a total of $O(d^{3/2}\mathrm{SK}(n,q))$. Finally, it takes at most $O(d^{3/2})$ additions in $\mathbb{L}$ to compute the sum $\sum_{i=0}^{d^*-1} c^{(i)}$. $\square$

When $d < n$, the algorithm of Puchinger & Wachter-Zeh has better bit complexity than the Caruso & Le Borgne algorithm, which is discussed next.

The Caruso & Le Borgne Algorithm

In [14], the authors gave two main algorithms for multiplying skew polynomials. The first of which is a Monte Carlo for the "large degree case" when the degree d is larger then the order of the automorphism $g = \mathrm{ord}(\sigma)$. The second is a deterministic algorithm when the sum of the degrees is less than g. While the latter is certainly of theoretical interest, for our purposes no such relation will be known so we will proceed exclusively with a discussion of the former. As the algorithm is quite technical, we will proceed with an outline of the procedure and a statement of its complexity.

Recall that L is an extension of $\mathbb{F}_q$ of order n, and for this setting we let σ have order g. Fix an extension K of $\mathbb{F}_q$ of order m and let $\mathbb{L}' = \mathbb{L} \otimes_K K$. The algorithm proceeds by computing the product $a \cdot b \bmod Z_i(\tau^g)$ for t irreducible polynomials $Z_1, \dots, Z_t \in \mathbb{F}_q[T]$, and reconstructs the product in $\mathbb{L}\{\tau\}$ via a Chinese remainder theorem to compute $a \cdot b \bmod Z_1 \cdots Z_t(\tau^g)$. The Z_i are constructed as minimal polynomials of elements $\lambda_i \in K$ such that $\lambda_i = N_{\mathbb{L}'/K}(\overline{\lambda}_i)$ for $\overline{\lambda}_i \in \mathbb{L}'$. For a choice of extension K with $\ell = [K : \mathbb{F}_q]$ such that $\frac{8d}{\ell g}\big(\frac{2d}{\ell g} + 1\big) \leq \ell q^\ell$, then $\ell \in O(\log d + \log g)$ and for any choice of $\{\lambda_i\}_{i=1}^{\lfloor \frac{2d}{\ell g} \rfloor}$, the probability that one can reconstruct the product in $\mathbb{L}\{\tau\}$ via the CRT is at least $\frac{1}{2}$ [14, Lemma 2.10, Theorem 2.11]. Multiplication in $\mathbb{L}\{\tau\}/Z_i(\tau^g)$ is reduced to multiplication in $\mathbb{L}'\{\tau\}/(\tau^g - \lambda_i)$ via the isomorphism $1 \otimes \mathrm{id} : \mathbb{L} \to K \otimes \mathbb{L}$ acting coefficient-wise and sending $\tau^g \mapsto a$.

Multiplication modulo $\tau^g - a$ can be lifted to multiplication modulo $\tau^g - 1$ via an isomorphism sending $a(\tau) \mapsto a(\overline{\lambda}\tau)$. To complete the picture, and give an algorithm for modular multiplication in $\mathbb{L}\{\tau\}/(\tau^g - 1)$, we first require the following proposition.

Proposition 1. *[14, Prop. 1.6] Let T be a commutative variable, and fix a normal basis $b_0, \ldots, b_{n-1}$ for $\mathbb{L}/\mathbb{F}_q$ and let $b = \sum_{i=0}^{n-1} b_i T^i$. Further let $a = \sum_{i=0}^{n-1} a_i \tau^i \in \mathbb{L}\{\tau\}$ and set $a(T) = \sum_{i=0}^{n-1} a_i T^i$, $c_i = a(b_i)$ and $c(T) = \sum c_i T^i$. Then*

$$c(T) = a(T)b(T) \mod T^n - 1$$

The above proposition reduces computing the action of a on a normal basis to commutative polynomial multiplication with the inputs reduced modulo $T^n - 1$. Given any two $a, a' \in \mathbb{L}[T]/(T^n - 1)$, the product aa' can be computed as follows:

1. Compute $c(T) = a(T)b(T)$, $c'(T) = a'(T)b(T)$.
2. Extract from c, c' the matrices M, M' respectively for the action of a, a' on $\mathbb{L}/\mathbb{F}_q$ where the domain uses the normal basis and the co-domain uses the standard basis.
3. Compute MBM' where B is the change of basis matrix from the standard basis to the normal basis. This gives the action of the product aa' on $\mathbb{L}/\mathbb{F}_q$.
4. Given the product MBM' compute the corresponding $\hat{c}(T)$ such that $\hat{c}b^{-1}(T) = a(T)a'(T)$ and extract the coefficients of aa'

After performing $O(d/tg)$ modular multiplications in $\mathbb{L}\{\tau\}/(Z_i(\tau^g))$, the cost of recovery via the CRT is $O(dg)$ field operations in $\mathbb{F}_q$. Via the reductions described above, each multiplication modulo $Z_i(\tau^g)$ reduces to a single instance of multiplication in $\mathbb{L}\{\tau\}/(\tau^g - 1)$. This reduction, as well as the multiplication itself, costs $O(g^\omega)$ $\mathbb{F}_q$-operations. When $d > g$, the complexity of the entire procedure can be bounded by $O(dg^{\omega-1}(\log d + \log g))$ $\mathbb{F}_q$-operations; in the context of the Frobenius endomorphism this gives a bit complexity of $\mathrm{SM}(d, n, q) = dn^{\omega-1}(\log d + \log n) \log q$. Moreover, there is a one-time cost of constructing a normal basis for $\mathbb{L}$ over $\mathbb{F}_q$, adding $n^{\omega_2/2} \log^{1+o(1)} q + n^{1+o(1)} \log^{2+o(1)} q$.

2.4.2 Euclidean Division

As alluded to in section §2.4, the ring $\mathbb{L}\{\tau\}$ is a right (and left) Euclidean domain and therefore admits a non-commutative analog of the Euclidean algorithm [60].

Operation 4 (Right Euclidean Division)**.** *Given skew-polynomials a, b, compute skew-polynomials ω, ρ such that $\deg_\tau(\rho) < \deg_\tau(b)$ and $a = \omega b + \rho$. In this case we write* $\mathrm{RD}(a, b) = (\omega, \rho)$*.*

Elementary division problems concerning skew polynomials were first studied in [60], with the first thorough algorithmic treatment, including the first non-trivial factorization scheme for skew polynomials, given in [35]. In [14] an algorithm for right Euclidean division of a degree d_1 skew polynomial a by a degree d_2 skew polynomial b with complexity in $O(\mathrm{SM}(d_1, n, q) \log d_1)$. The procedure is based on a classical algorithm for Euclidean division on commutative polynomials seen in [30].

Theorem 2.4.3. *The right Euclidean quotient and remainder* $\mathrm{RD}(a, b)$ *of two skew polynomials a, b of degree at most d can be computed using $O(\mathrm{SM}(d, n, q) \log d)$ bit operations [64].*

We sketch the procedure here. Define the pseudo-inversion operator of order δ on $\mathbb{L}\{\tau\}_\delta$:

$$T_\delta\left(\sum_{i=0}^{\delta} a_i \tau^i\right) = \sum_{i=0}^{\delta} a_{\delta-i} \tau^{-i} \tag{2.1}$$

For a skew polynomial $\rho = \sum_{j=0}^{d} \rho_j \tau^j$, let $\rho^{(i)} = \sum_{j=0}^{d} \sigma^i(\rho_j) \tau^j$. It is fairly straightforward to check that T_δ is linear and satisfies the relation $T_{d_1+d_2}(ab) = T_{d_1}(a) T_{d_2}(b^{(\delta_1)})$. Let $\overline{\omega} = \sum_{i=0}^{\infty} \overline{\omega}_i \tau^i$ be such that $\overline{\omega} T_{d_2}(b^{d_1-d_2}) = 1$. From the preceding product formula, we have that $T_{d_1}(a) = T_{d_1-d_2}(\omega) T_{d_2}(b) + T_{d_1}(\rho)$. Let $\overline{\omega}^{[d_1-d_2]} = \sum_{i=0}^{d_1-d_2-1} \overline{\omega}_i \tau^i$; and observe that $T_{d_1}(a) \overline{\omega}^{[d_1-d_2]} b - T_{d_1}(a)$ must have degree at least $d_1 - d_2$, implying that $T_{d_1-d_2}(\rho) = T_{d_1}(a) \overline{\omega}^{[d_1-d_2]} b - T_{d_1}(a)$. We can compute $\overline{\omega}^{[d_1-d_2]}$ using Newton iteration in $O(\log d_1)$ steps, each of which has cost at most $O(\mathrm{SM}(d_1, n, q))$.

2.4.3 Minimal Subspace Polynomial & Multipoint Evaluation

Recall that we have a canonical identification $\xi : \mathbb{L}\{\tau\} \to \mathrm{End}_{\mathbb{F}_q}(\mathbb{L})$. The *kernel* of a skew polynomial ρ is the set:

$$\ker(\rho) = \{a \in \mathbb{L} \mid \rho(a) = 0\}.$$

Observe that $\ker(\rho)$ is an $\mathbb{F}_q$-linear subspace of $\mathbb{L}/\mathbb{F}_q$ and $\dim \ker(\rho) \leq \min(\deg \rho, n)$, with the latter fact following from the observation that if $\rho = \sum_{i=0}^{d} a_i \tau^i$, then $a \in \ker(\rho)$ if and only if it is a root of the linearized polynomial $\sum_{i=0}^{d} a_i x^{q^i}$. Let $V \subset \mathbb{L}$ be a linear subspace of dimension $m \leq n$ over $\mathbb{F}_q$. The *minimal subspace polynomial* $\mathrm{MSP}(V)$ is the monic skew-polynomial of lowest degree whose kernel contains V. The MSP of a linear subspace is unique and has degree d if V has dimension d. Consequently, a monic degree d skew-polynomial is equal to $\mathrm{MSP}(V)$ for some subspace V if and only if it has a set of d linearly independent roots.

Problem 2.4.4 (Minimal Subspace Polynomial)**.** *Given a linear subspace $V \subset \mathbb{L}/\mathbb{F}_q$ with a basis $v_1, \ldots, v_d \in \mathbb{L}$, determine its minimal subspace polynomial,* $\mathrm{MSP}(V) = \mathrm{MSP}(\{v_1, \ldots, v_d\})$.

Given a set of s elements $v_1, \ldots, v_s$ and a skew polynomial ρ we may wish to simultaneously compute the evaluations $\rho(v_1), \ldots, \rho(v_s)$. This leads us to the *multipoint evaluation* problem.

Problem 2.4.5 (Multipoint Evaluation)**.** *Given a set of s points $v_1, \ldots, v_s$ and $\rho \in \mathbb{L}\{\tau\}$, compute the set* $\mathrm{MPE}(\rho; v_1, \ldots, v_s) = \{\rho(v_1), \ldots, \rho(v_s)\}$.

In the case of Multipoint evaluation, it suffices to compute the evaluation on a maximal linearly independent subset of $v_1, \ldots, v_s$. Two mutually recursive algorithms were proposed in [64] to solve both problems. For details on these, see algorithms 1 and 2.

A detailed complexity analysis for these mutually recursive algorithms was given in [64]. We will let $\mathrm{MSP}^*(d)$ denote the complexity of computing the minimal subspace polynomial on a space of dimension d, and $\mathrm{MPE}^*(d, s)$ the complexity of computing the multipoint evaluation of a degree d skew polynomial on s points. Then the cost of both algorithms can be expressed in terms of the following recurrences:

$$\mathrm{MSP}^*(d) = 2\mathrm{MSP}^*(d/2) + \mathrm{MPE}^*(d/2, d/2) + \mathrm{SM}(d/2, n, q)$$
$$\mathrm{MPE}^*(d, s) = 2\mathrm{MSP}^*(s/2) + 2\mathrm{MPE}^*(d/2, s/2) + \mathrm{SM}(s, n, q) \log s.$$

Algorithm 1 Minimal Subspace Polynomial [64]

Input A basis $\{v_1, \ldots, v_d\}$ for $V \subset \mathbb{L}$

Output A monic skew-polynomial $\rho \in \mathbb{L}\{\tau\}_d$ of degree d such that $\rho(v_i) = 0$ for each $1 \leq i \leq d$.

```
1: if $d = 1$ then
2:     $\rho \leftarrow 1$ if $v_1 = 0$
3:     $\rho \leftarrow \tau - v_i^{q-1}$ otherwise
4: else
5:     $a \leftarrow \mathrm{MSP}(v_1, \ldots v_{d/2})$
6:     $\{v'_{d/2+1}, \ldots, v'_d\} \leftarrow \mathrm{MPE}(a; v_{d/2+1}, \ldots, v_d)$
7:     $b \leftarrow \mathrm{MSP}(v'_{d/2+1}, \ldots, v'_d)$
8:     $\rho \leftarrow ba$
9: end if
```

Algorithm 2 Multipoint Evaluation [64]

Input A skew polynomial $\rho \in \mathbb{L}\{\tau\}$ of degree d, and a set of linearly independent evaluation points $\{v_1, \ldots, v_s\} \subset \mathbb{F}_{q^n}$.

Output The set $\{\rho(v_1), \ldots, \rho(v_s)\} \subset \mathbb{F}_{q^n}$

```
1: if $\deg_\tau(\rho) = 1$ then
2:     Return$\{\rho(v_1), \ldots, \rho(v_s)\}$
3: else
4:     $a \leftarrow \mathrm{MSP}(v_1, \ldots v_{s/2})$
5:     $b \leftarrow \mathrm{MSP}(v_{s/2+1}, \ldots v_s)$
6:     $(\omega_a, \rho_a) \leftarrow \mathrm{RD}(\rho, a)$
7:     $(\omega_b, \rho_b) \leftarrow \mathrm{RD}(\rho, b)$
8:     $\{\rho(v_1), \ldots, \rho(v_{s/2})\} \leftarrow \mathrm{MPE}(\rho_a; v_1, \ldots v_{s/2}))$
9:     $\{\rho(v_{s/2+1}), \ldots, \rho(v_s)\} \leftarrow \mathrm{MPE}(\rho_b; v_{s/2+1}, \ldots v_s))$
10: end if
```

By memoizing the output of MSP at line 5, the call to MSP at line 4 of algorithm 2 can be avoided, and the recursive expression for MPE becomes

$$\mathrm{MPE}^*(d,s) = \mathrm{MSP}^*(s/2) + 2\mathrm{MPE}^*(d/2,s/2) + \mathrm{SM}(s,n,q)\log s.$$

Let $\mathfrak{d} = \max(s,d)$; both expressions are simultaneously bounded by a complexity class $C(\mathfrak{d},n,q)$ satisfying the recurrence

$$C(\mathfrak{d},n,q) = 3C(\mathfrak{d}/2,n,q) + O(\mathrm{SM}(\mathfrak{d},n,q)\log\mathfrak{d})$$

By assuming the complexity of the Puchinger-Wachter-Zeh algorithm $\mathrm{SM}(d,n,q) = O(d^{\omega_2/2}\mathrm{Mul}(n,q) + d^{3/2}\mathrm{SK}(n,q))$ and using the master theorem we can obtain a complexity of

$$\begin{aligned} C(\mathfrak{d},n,q) &= \tilde{O}(\mathrm{SM}(\mathfrak{d},n,q) + \mathfrak{d}^{\log_2(3)}n\log q) \\ &= \tilde{O}(\mathfrak{d}^{\max(\log_2(3),\omega_2/2)}\mathrm{Mul}(n,q)\log\mathfrak{d} + \mathfrak{d}^{\log_2(3)}\mathrm{SK}(n,q)\log\mathfrak{d}) \\ &\approx \tilde{O}(\mathfrak{d}^{\max(\log_2(3),\omega_2/2)}n\log q) \end{aligned}$$

bit operations. We will discuss an alternate approach to both the multipoint evaluation and minimal subspace polynomial problems in section §7.1.

2.5 Valuations, Places, and Limits

Valuations, valued fields, and places play an important role in algebraic number theory, the definition of Drinfeld modules, and their cohomology theories.

Definition 2.5.1. *Given a field $\mathbb{F}$ and an abelian group G equipped with a total ordering $<$, a* ***valuation*** *on $\mathbb{F}$ is a map $v : \mathbb{F} \to G \cup \{\infty\}$ satisfying:*

- *$v(a) = \infty$ if and only if $a = 0$*
- *$v(ab) = v(a) + v(b)$*

- $v(a+b) \geq \min(v(a), v(b))$

Furthermore, let g^* be an order-reversing group homomorphism $g^* : G \to (\mathbb{R}\backslash\{0\}, \times)$ with the extension $g^*(\infty) = 0$. Then the valuation v induces a *norm* $\|\cdot\|_v$ on $\mathbb{F}$ given by $\|a\|_v = g^*(v(a))$. We then define $\mathbb{F}_v$ to be the completion of $\mathbb{F}$ with respect to $\|\cdot\|_v$. Two valuations v_1, v_2 are said to be equivalent if there is an order preserving automorphism $g : G \to G$ such that $v_1(a) = g(v_2(a))$. In this case, the norms induced by v_1 and v_2 are equivalent and $\mathbb{F}_{v_1} \cong \mathbb{F}_{v_2}$.

Definition 2.5.2. *A **place** of a field $\mathbb{F}$ is an equivalence class of valuations.*

In the next section, we will discuss a classical example of valuations and their corresponding places.

2.5.1 The p-adic Numbers

The most famous class of examples of valuations are the so-called *p-adic* valuations $v_p : \mathbb{Q} \to \mathbb{Z} \cup \{\infty\}$ on $\mathbb{Q}$. These are defined for a fixed prime p as follows:

- $v_p(0) = \infty$
- $v_p(x) = \max(n : p^n | x)$ for all $x \in \mathbb{Z}$
- If $x = \frac{a}{b}$ for $a, b \in \mathbb{Z}$, then $v_p(x) = v_p(a) - v_p(b)$

Let $|\cdot|_p$ denote the norm induced by v_p under the mapping $g^*(z) = p^{-z}$ for all $z \in \mathbb{Z}$, and let $\mathbb{Q}_p$ denote the completion of $\mathbb{Q}$ with respect to $|\cdot|_p$. We may then construct *p-adic integers* $\mathbb{Z}_p$ in a number of equivalent ways.

Definition 2.5.3. *The set $\mathbb{Z}_p$ of p-adic integers may be defined in any of the following ways:*

1. $\{x \in \mathbb{Q}_p \mid |x|_p \leq 1\}$

2. *the completion of $\mathbb{Z}$ with respect to $|\cdot|_p$*

3. *the set of formal power series*
$$\sum_{i=0}^{\infty} a_i p^i$$
with entries $a_i \in \{0, 1, \ldots, p-1\}$.

Definition 3 will be of particular interest to us later when generalizing p-adic constructions to general rings. A given p-adic integer is more typically written in its *p-adic expansion* as $\ldots a_2 a_1 a_0$. There is a canonical inclusion $\mathbb{Z} \hookrightarrow \mathbb{Z}_p$, given by, for any integer $\ell \in \mathbb{Z}$, the sequence satisfying, $\sum_{j=0}^{i-1} a_j p^j \equiv \ell \bmod p^i$ for all $i \geq 1$.

Let $q = p^n$, fix an irreducible $h \in \mathbb{F}_p[x]$, and set $\hat{h}$ to be a lift of h to $\mathbb{Z}_p$. The object $\mathbb{Q}_q = \mathbb{Q}_p[x]/(\hat{h}(x))$, is unique up to isomorphism regardless of the choice of lifting. When $\mathbb{F}_q$ is given as $\mathbb{F}_q = \mathbb{F}_p[x]/(h(x))$, we can then define a lifting $\mathbb{F}_p[x]/(h(x)) \to \mathbb{Q}_p[x]/(\hat{h}(x))$ by the usual mapping of coefficients $\{0, \ldots, p-1\}$ into $\mathbb{Z}_p$.

A similar construction can be carried out for polynomial rings $\mathbb{F}[x_1, \ldots, x_k]$. Namely, choose a prime ideal I, define $v_I(x) = \max(n : x \in I^n)$, and extend v_I to $\mathbb{F}(x_1, \ldots, x_k)$ in the usual manner. If $p > 0$ is the characteristic of the field $\mathbb{F}$, then we can obtain a norm as before by letting $\|x\|_I = p^{-v_I(x)}$, and each choice of prime ideal I defines a distinct place. For multivariate polynomial rings there are additional places, defined by the valuation $v_\infty(x) = \deg(x)$ where deg may denote either the degree in a fixed variable x_i or the total degree.

2.5.2 Inverse Limits for Rings and Modules

The concept of limits in category theory provide a vast generalization of many constructs used throughout algebra, including disjoints unions, cartesian products, direct sums, etc. For our purposes, we will make use of a heavily simplified definition of a particular type of limit, known as an *inverse limit.*

Definition 2.5.4. *Let $\{H_i\}_{i\in\mathbb{N}}$ be an* ***inverse system****; that is, a family of rings such that for each $i \leq j$ we have ring homomorphisms $f_{i,j} : H_j \to H_i$ such that*

- *$f_{i,i}$ is the identity map on H_i for all i*
- *for all $i \leq j \leq k$, $f_{i,j} \circ f_{j,k} = f_{i,k} : H_k \to H_i$.*

*The we define the **inverse limit** of $\{H_i\}_{i \in \mathbb{N}}$ to be the ring*

$$\varprojlim_i H_i = \left\{ (h_j)_{j \in \mathbb{N}} \in \prod_{i \in \mathbb{N}} H_i \mid f_{i,j}(h_j) = h_i \forall i \leq j \right\}$$

That is, $\varprojlim H_i$ is the set of sequences $(h_j)_{j \in \mathbb{N}}$ such that $h_j \in H_j$ and $f_{i,j}(h_j) = h_i$ for any pair $i \leq j$; this is a ring with the usual operations defined entrywise. The inverse limit also requires the existence of *projection maps* $\pi_i : H \to H_i$ which are ring homomorphisms such that $\pi_i = f_{i,j}\pi_j$ for all $i \leq j$; under the construction given in definition 2.5.4, the co-ordinate functions $\pi_i((h_j)_{j \in \mathbb{N}}) = h_i$ satisfy these requirements.

We will take care to note that the inverse limit construction can be carried out for more general algebraic objects and categories, though we will require it only for rings and modules. For inline typesetting we may omit the index of the inverse limit. This construction also gives another way of constructing the p-adic integers, namely as the inverse limit $\mathbb{Z}_p = \varprojlim \mathbb{Z}/(p^i)$. Moreover, this approach can be easily generalized to arbitrary rings.

Definition 2.5.5. *Let R be a ring, and let I be a prime ideal of R. The **I-adic completion** of R with respect to I is the ring*

$$R_I = \varprojlim_k R/I^k$$

The elements of R_I can be viewed as convergent series in the I-adic norm $\|\cdot\|_I$. Now suppose we are given a module H over a ring R, and let I be any prime ideal of R. We can define the quotient modules $H_i = H/I^iH$ which is also a module over R/I^i, and we have maps $f_{i,j} : H_j \to H_i$ when $i \leq j$ obtained by reducing elements of H_j to their equivalence classes modulo I^iH. We can then define the inverse limit of the modules $H = \varprojlim H_i$ in an identical manner to the construction for rings in definition 2.5.4, with the additional remark that H is a module over R_I where the ring action can be taken co-ordinate wise. More explicitly, if $(r_i)_{i \in \mathbb{N}} \in R_I$ and $(h_i)_{i \in \mathbb{N}}$ then $(r_i)_{i \in \mathbb{N}} * (h_i)_{i \in \mathbb{N}} = (r_i * h_i)_{i \in \mathbb{N}}$.

Chapter 3

Elliptic Curves

3.1 Definitions

The primary goal of this section will be to review some of the classical theory of Elliptic curves, and in particular the approaches used for computing the characteristic polynomial of the Frobenius endomorphism. As Drinfeld modules are often viewed as part of the "function field" analogy with classical number fields, with rank 2 Drinfeld modules being a direct parallel to elliptic curves, this section is intended to motivate the approaches used later on in the text by describing their inspiration from the elliptic curve setting. For those interested in further readings on elliptic curves, we recommend [71], [69], and [70].

We will recall some basic concepts from algebraic geometry, which can be reviewed in most standard references on the topic such as [38]. We begin by supposing $\mathbb{F}$ is an algebraically closed field, an *affine variety* over $\mathbb{F}^k$ is a subset $V \subset \mathbb{F}^k$ whose elements are the zeros of all members of an ideal $S \subset \mathbb{F}[x_1, \dots, x_k]$. We will define *projective k-space* $\mathbb{P}^k(\mathbb{F})$ to be the set whose elements are the 1-dimensional subspaces of $\mathbb{F}^{k+1}$. More explicitly, we can view $\mathbb{P}^k(\mathbb{F})$ as the set of equivalence classes in $\mathbb{F}^{k+1}$ under the equivalence relation $s \sim s'$ if there exists $\lambda \in \mathbb{F}$ such that $s = \lambda s'$. A polynomial in $\mathbb{F}[x_1, \dots, x_k]$ is *homogeneous* if all its monomial terms have the same total degree. A *projective variety* is a subset $V \subset \mathbb{P}^k(\mathbb{F})$ whose elements are the zeros of all members of an ideal $S \subset \mathbb{F}[x_1, \dots, x_{k+1}]$

generated by homogeneous polynomials. Note that we can canonically embed homogeneous elements of $\mathbb{F}[x_1, \ldots, x_{k+1}]$ into $\mathbb{F}[x_1, \ldots, x_k]$ via the evaluation map $x_{k+1} = 1$; this allows us to explicitly describe projective varieties as subsets of k dimensional spaces vanishing at the members of an ideal generated by ordinary k-variate polynomials. A function $f : V \to V'$ is *regular* if it can be expressed component-wise as polynomial functions. A variety V is *abelian* if it has a commutative group structure on its points that is "given by regular maps". That is:

- there is a regular map $+ : V \times V \to V$ satisfying the usual axioms of a group operation, such that $a + b = b + a$ for all $a, b \in V$
- there is an element $\mathrm{id} \in V$ such that $a + \mathrm{id} = a$ for all $a \in V$
- there is a regular map $- : V \to V$ such that $a + (-a) = \mathrm{id}$ for all $a \in V$.

The map $+$ is sometimes referred to as the *group law* on the variety.

We will typically choose to work with $\mathbb{F} = \overline{\mathbb{F}}_q$ and consider the restrictions of curves to sub-fields as necessary. In the algebraically closed setting, Hilbert's *Nullstellensatz* tells us that affine varieties $V \subset \mathbb{F}^k$ correspond to radical ideals $V(I)$ of $\mathbb{F}[x_1, \ldots, x_k]$ and that this correspondence is bijective. Then the *regular functions* from V to $\mathbb{F}$ are the elements of $\mathbb{F}[x_1, \ldots, x_k]/V(I)$. A similar correspondence concerning homogeneous ideals exists for projective varieties.

Definition 3.1.1. *Let $\mathbb{F}$ be an algebraically closed field whose characteristic is different from 2 or 3. An **elliptic curve** E over $\mathbb{F}$, denoted $E/\mathbb{F}$, is a smooth abelian projective variety over $\mathbb{P}_2(\mathbb{F})$ whose elements are the zeros of a Weierstrass equation*

$$y^2 = x^3 + ax + b$$

for fixed $a, b \in \mathbb{F}$ with $4a^3 + 27b^2 \neq 0$.

While a more general definition of elliptic curves does exist, in particular for the "small" characteristic case, this definition will be sufficient for the purposes of this work. For a subfield $\mathbb{F}' \subset \mathbb{F}$, we will set $E/\mathbb{F}' \subset E/\mathbb{F}$ to be the set of points invariant under the natural action of $\mathrm{Gal}(\mathbb{F}/\mathbb{F}')$ on $\mathbb{P}_2(\mathbb{F})$. As an abelian variety, any elliptic curve has a group law defined on it; addition of points can be defined explicitly as follows: the projective point at infinity acts as the identity of the group. Otherwise, we have the *doubling* and *addition* formulae. Set

$$d_c(x,y) = \frac{3x^2+a}{2y}$$

Then for any $(x,y) \in E$, the double of any point can be written as

$$2(x,y) = (d_c(x,y)^2 - 2x, 3d_c(x,y)x - d_c(x,y)^3 - y)$$

Otherwise, for $x_1 \neq x_2$, set

$$a_c = a_c(x_1,y_1,x_2,y_2) = \frac{y_2-y_1}{x_2-x_1}$$

Addition of two distinct points in E can then be computed as follows:

$$(x_1,y_1)+(x_2,y_2) = (a_c^2 - x_1 - x_2, a_c(2x_1 - a_c^2 + x_2) - y_1)$$

We will fix $O \in E$ to be the identity element for the group operation on $E/\mathbb{F}$. For a positive integer t, we let $t * P = \overbrace{P + P + \ldots + P}^{t \text{ times}}$, and we define $-P \in E$ to be the point such that $P + (-P) = O$. We then define the *t-torsion* of an elliptic curve E to be the set

$$E[t] = \{P \in E \mid t * P = O\}.$$

Let $\mathrm{Hom}(E_1, E_2)$ denote the set of group morphisms from $E_1/\mathbb{F} \to E_2/\mathbb{F}$ that are also regular maps. We will use the term *endomorphism* to refer to elements of $\mathrm{Hom}(E,E)$. Let $\mathbb{F}$ be an algebraically closed field with positive characteristic p and suppose $E/\mathbb{F}$ is an elliptic curve defined over $\mathbb{F}$ whose Weierstrass equation has coefficients contained in $\mathbb{F}_q \subset \mathbb{F}$ with $q = p^e$. The *Frobenius morphism* on E is the map $F : E \to E'$ given by

$$\begin{aligned} F : E &\to E' \\ (x, y) &\mapsto (x^p, y^p) \end{aligned}$$

If $E/\mathbb{F}$ has Weierstrass equation $y^2 = x^3 + ax + b$, then $E'/\mathbb{F}$ is defined by the equation $y^2 = x^3 + a^p x + b^p$. When $a, b \in \mathbb{F}_q$, entry-wise exponentiation by q defines the (q-order) *Frobenius endomorphism* $F_q = F^e$ on E; the fixed points of this endomorphism are exactly the points contained in $E/\mathbb{F}_q$. The endomorphism F_q satisfies a degree two polynomial with integer coefficients $x^2 - c_1 x + c_2 \in \mathbb{Z}[x]$ known as the *characteristic polynomial* of the Frobenius. That is: $F_q^2(P) - c_1 * F_q(P) + c_2 * P = O$ for all $P \in E/\mathbb{F}$. The Frobenius endomorphism, along with its characteristic polynomial, plays an important role in point counting problems on algebraic varieties owing to conjectures and results originating in the work of Hasse and Weil [67] [19], and is a special case of a more general connection between zeta functions on varieties and schemes and point counting problems. In the next section, we will give a brief overview of these particular point counting problems in the setting of elliptic curves and make explicit the role played by the Frobenius endomorphism.

3.2 Point Counting

Given an elliptic curve $E/\mathbb{F}$ we will let E_q denote the set of points on the curve whose coordinates lie entirely in $\mathbb{F}_q$.

Operation 5. *Given an elliptic curve $E/\mathbb{F}$ and a positive power of the field characteristic $q = p^e$, compute $|E_q|$.*

Now we will assume that $E/\mathbb{F}$ is defined by a Weierstrass equation whose coefficients are all contained inside a finite field $\mathbb{F}_q$. The role that the q-order Frobenius endomorphism plays in computing the number of points in E_q, as well as bounds for the amount of computational work required to compute $|E_q|$, are consequences of theorems due to Hasse and Weil [69, Thm. 1.1].

Theorem 3.2.1 (Hasse)**.** *Let $q = p^e$. Let $E/\mathbb{F}$ be an elliptic curve over an algebraically closed field $\mathbb{F}$ of characteristic p whose Weierstrass equation has coefficients contained in $\mathbb{F}_q$. Then*

$$|q + 1 - |E_q|| \leq 2\sqrt{q}$$

A powerful consequence of the general Weil conjectures gives an explicit connection between the characteristic polynomial of F_q and $|E_q|$.

Theorem 3.2.2 (Weil's theorem for elliptic curves)**.** *Let $E/\mathbb{F}$ be an elliptic curve be as defined in theorem 3.2.1. Set $h = q + 1 - |E_q|$ and let F_q be the q-order Frobenius acting on $E/\mathbb{F}$. Then F_q satisfies the equation*

$$F_q^2 - hF_q + q = 0. \tag{3.1}$$

3.3 Schoof's Algorithm

Schoof [65] gave the first polynomial time algorithm, in $\log q$, for computing $|E|$ by finding $|E| \bmod p_i$ for distinct primes p_i such that $\prod_i p_i > 4\sqrt{q}$. In particular, we know that for $(x, y) \in E[p]$, $p * (x, y) = 0$ and so we have $t * (x, y) = (t \bmod p) * (x, y)$, which allows us to reduce the characteristic polynomial modulo p to obtain

$$(x^{q^2}, y^{q^2}) + (q \bmod p)(x, y) = (h \bmod p)(x^q, y^q)$$

To compute the characteristic polynomial of F on $E[p]$, Schoof introduced the so-called *division polynomials*, which allow one to efficiently encode entire torsion subgroups via polynomials $\psi_p(x, y)$ whose zero sets on E are exactly $E[p]$. Concretely, the division polynomials are polynomials in $\mathbb{Z}[x, y, a, b]$ corresponding to elliptic curves with Weierstrass equation $y^2 = x^3 - ax + b$, and can be constructed recursively as follows:

$$\begin{aligned}
\psi_0 &= 0 \\
\psi_1 &= 1 \\
\psi_2 &= 2y \\
\psi_3 &= 3x^4 + 6ax^2 + 12bx - a^2 \\
\psi_4 &= 4y(x^6 + 5ax^4 + 20bx^3 - 5a^2x^2 - 4abx - 8b^2 - a^3) \\
&\vdots \\
\psi_{2m+1} &= \psi_{m+2}\psi_m^3 - \psi_{m-1}\psi_{m+1}^3 \\
\psi_{2m} &= \frac{\psi_m}{2y}(\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2)
\end{aligned}$$

Lemma 3.3.1. *[65] The points $(x, y) \in E$ satisfying $\psi_p(x, y) = 0$ are exactly $E[p]$.*

This leads us to the original version of Schoof's algorithm:

- Fix primes $p_1, \ldots, p_t$ such that for all i, $p_i \nmid q$ and $\prod_{i=1}^t p_i > 4\sqrt{q}$,
- Solve for h_i satisfying:

$$(x^{q^2}, y^{q^2}) + (q \bmod p_i)(x, y) = h_i(x^q, y^q).$$

- Return the unique h such that $h \bmod p_i = h_i$ for all i.

The original version of Schoof's algorithm, using modern computational techniques, had a runtime of $\tilde{O}(\log^5 q)$ bit operations. Later improvements due Elkies and Atkins replace the division polynomials with *modular polynomials* $\Phi_p(X, Y)$ whose roots correspond to pairs of so-called *p-isogenous curves.* Their improvements leveraged information about the splitting behaviour of the characteristic polynomial by alternately chosing primes p_i based on the splitting behaviour of the characteristic polynomial modulo p_i. These improvements gave a runtime of $\tilde{O}(\log^4 q)$ bit operations.

3.4 Kedlaya's Algorithm

While polynomial in $\log q$, Schoof's algorithm still faces some notable drawbacks, including the still fairly large exponent in the complexity in $\log q$, as well as exponential complexity in the genus of the curve when the algorithm is generalized to hyperelliptic curves. For a fixed prime p such that $q = p^e$, later algorithms, notably Satoh's and Kedlaya's, had improved complexity in terms of e at the cost of no longer being polynomial in $\log p$.

Kedlaya's algorithm [49] for computing characteristic polynomials of hyperelliptic curves uses a variant of crystalline cohomology, that of Monsky-Washnitzer, and for fixed p has a complexity of $O(g^{4+o(1)}e^{3+o(1)})$ [25]. The core idea of Kedlaya's algorithm is to compute the matrix of the lifting of the Frobenius endomorphism to a particular subspace of the Monsky-Washnitzer cohomology, whose coefficients, lying in $\mathbb{Z}_q$, can be computed to a finite precision. We will now introduce the machinery required to make this description more precise.

We will note that this next section is quite heavy technically and the machinery described here is not required for the main results. This section is included mainly to provide a background description of the inspiration for the crystalline algorithm for computing the characteristic polynomial of the Frobenius endomorphism of a Drinfeld module described later in this work. The reader may therefore feel free to skip this section without impacting their understanding of the rest of the content of this work.

3.4.1 Monsky-Washnitzer Cohomology

For readers interested in exploring crystalline or the more general setup of p-adic cohomology, we recommend [6]. We will give an explicit algebraic description of the crystalline cohomology on an elliptic curve by way of its construction via the "algebraic de Rham" cohomology. We recall our initial setting of an elliptic curve $E/\mathbb{F}_q$ with equation $y^2 = f(x) = x^3 + ax + b$. First, we observe that the Frobenius automorphism on $\mathbb{F}_q$ can be lifted to a Frobenius automorphism on $\mathbb{Z}_q$. We can lift to a curve $\hat{E}/\mathbb{Q}_q$ using a choice of lift $\hat{f} = x^3 + \hat{a}x + \hat{b} \in \mathbb{Z}_q[x]$. The ring of regular functions on $E_0 = E - \{(x, y) \mid y = 0\}$ is given by

$$A_0 = \mathbb{F}_q[x, y, y^{-1}]/(y^2 - f(x))$$

The crystalline cohomology is typically defined in terms of the de Rham complex associated to a canonical lifting of A to regular functions on the lift $\hat{E}_0/\mathbb{Q}_q$. In the case of Monsky-Washnitzer cohomology, we instead consider the so-called *overconvergent* functions:

$$A_0^\dagger = \left\{ \sum_{i\in\mathbb{Z}} (a_{0,i} + a_{1,i}x + a_{2,i}x^2)y^i \mid \liminf_{|j|\to\infty} |a_{j,i}|_p/|j| > 0 \right\}$$

The standard derivation on A_0 can be obtained by setting

$$2ydy = (3x^2 + a)dx$$

The first de Rham complex of the Monsky-Washnitzer cohomology is therefore given as

$$H^1_{MW}(E) = \ker\ d/dA_0^\dagger = A_0^\dagger \frac{dx}{2y}/dA_0^\dagger$$

The map $\iota : E \to E$ given by

$$\iota(x, y) = (x, -y)$$

induces an endomorphism on A_0 sending $y \mapsto -y$, and similarly on $H^1_{MW}(E)$. This map splits $H^1_{MW}(E)$ into a direct sum of eigenspaces $H^1_{MW}(E) = H^1_{MW}(E)^+ \oplus H^1_{MW}(E)^-$ corresponding to the eigenvalues ± 1 of ι. We then have the following lemma:

Lemma 3.4.1. *The set* $\mathcal{B} = \left\{\frac{dx}{y}, \frac{xdx}{y}\right\}$ *is a basis for* $H^1_{MW}(E)^-$.

3.4.2 Lifting the Frobenius

Working with basis $\mathcal{B}$, we will be able to explicitly compute the action of the Frobenius on $H^1_{MW}(E)^-$.

Lemma 3.4.2. *There exists a lifting of the Frobenius endomorphism F on E to $F^\dagger : A_0^\dagger \to A_0^\dagger$ satisfying the following relations:*

- $F^\dagger(x) = x^p$
- $F^\dagger(y)^2 = x^{3p} + \hat{a}^p x^p + \hat{b}^p$
- $F^\dagger(y)F^\dagger(y^{-1}) = 1$.

Then $F^\dagger$ lifts to a map $\mathcal{F}$ on $H^1_{MW}(E)^-$ whose action on the elements of the basis $\mathcal{B}$ is given by:

$$\mathcal{F}\left(\frac{dx}{y}\right) = \sum_{k=0}^{\infty} \binom{-1/2}{k} p^{k+1} \frac{F^\dagger(\hat{f}) - \hat{f}^p}{p} x^{p-1} y^{-2kp-p} dx$$

$$\mathcal{F}\left(\frac{xdx}{y}\right) = \sum_{k=0}^{\infty} \binom{-1/2}{k} p^{k+1} \frac{F^\dagger(\hat{f}) - \hat{f}^p}{p} x^{2p-1} y^{-2kp-p} dx$$

We can re-express $\mathcal{F}(\frac{dx}{y})$ as :

$$\mathcal{F}\left(\frac{dx}{y}\right) = \sum_{k=0}^{\infty} p^{k+1} c_k(x) \frac{dx}{y} \tag{3.2}$$

where $c_k(x) \in \mathbb{Q}_q(x)$ such that $p^{k_0} c_k(x) \in \mathbb{Z}_q(x)$ when $k_0 > \log_p(2k+1) + 1$ [25]. This expression can be rewritten in terms of the basis elements of $\mathcal{B}$ using the reduction algorithm of [25]. We then have the following theorem, which allows us to justify this entire computation.

Theorem 3.4.3. *[25, Thm. 5.3.2] The characteristic polynomial of F^e on E is equal to the characteristic polynomial of $\mathcal{F}^e$ on $H^1_{MW}(E)^-$. That is, it has the form*

$$X^2 - \hat{h}X + \hat{q}$$

Moreover, recalling that $|h| \leq 4\sqrt{q}$, we have that $|\hat{h}|_p \leq \frac{n}{2}\log_p 4$.

The goal then is to explicitly compute the the matrix of $\mathcal{F}^e$ via the lifting of $\mathcal{F}$, and extract the characteristic polynomial of this matrix. If $\mathcal{M}(\mathcal{F})$ is the matrix for $\mathcal{F}$ on $H^1_{MW}(E)^-$, then a matrix for $\mathcal{F}^e$ is given by

$$\mathcal{M}(\mathcal{F}^e) = \mathcal{M}(\mathcal{F})\mathcal{M}(\mathcal{F})^{\sigma} \cdots \mathcal{M}(\mathcal{F})^{\sigma^{e-1}},$$

where we recall that $\mathcal{M}(\mathcal{F})^{\sigma}$ denotes entry-wise application of σ when σ is an operator. The algorithm can be summed up as follows:

1. Fix a lift $\hat{E}/\mathbb{Q}_q$ of $E/\mathbb{F}_q$. Set precision $\pi \geq \frac{n}{2}\log_p 4$.
2. Compute the action of $\mathcal{F}$ on the basis $\mathcal{B}$ by computing the coefficient polynomials $c_k(x)$ up to k satisfying $k - \log_p(2k+1) - 1 > \pi$.
3. Using the reduction algorithm, extract the coefficients of the matrix $\mathcal{M}(\mathcal{F})$ with respect to $\mathcal{B}$.
4. Compute the matrix for $\mathcal{F}^e$ as $\displaystyle\prod_{i=0}^{e-1} \mathcal{M}(\mathcal{F})^{\sigma^i}$. Compute its characteristic polynomial and return the result.

The dominant computational step is item 4, which has a complexity of $e^{3+o(1)}$ bit operations. For more general curves of genus g, the analogous algorithm has a runtime using $g^{4+o(1)}e^{3+o(1)}$ bit operations.

Chapter 4

Drinfeld Modules

Drinfeld modules were first introduced by Vladimir Drinf'eld [22] as part of his proof of the Langlands conjecture for GL_2 over global function fields. Since then, Drinfeld modules and their generalizations have been recognized as important tools in the theory of function fields, playing a role analogous to that of elliptic curves. In line with this philosophy, a significant amount of theory, paralleling that of the number field case, has therefore been developed for Drinfeld modules. This has included questions of computational significance, given the important role that elliptic curves play in cryptography [42] [54], [74] and integer factorization [59]. The focus of this section will be to introduce the main definitions surrounding Drinfeld modules and associated objects, and cover some of the background theory relevant to understanding the characteristic polynomial computation.

4.1 Definitions

4.1.1 Background

We now introduce the main definitions and constructions, which can be found in the survey of Drinfeld modules of Deligne and Husemöller in [20]. Fix a smooth projective curve C defined over $\mathbb{F}_q$, and let A denote the ring of functions on C which are regular outside of a fixed place. The standard example of such a setting, and the one that we will focus on for the purpose of explicitly describing our algorithms, takes $C = \mathbb{P}^1 \subset \mathbb{P}^2$ and at the place $v_\infty = \deg$, in which case $A = \mathbb{F}_q[x]$. Let $K = \mathrm{Frac}(A)$ denote the fraction field of A; which in our standard case will be $\mathbb{F}_q(x)$.

Now let $\mathbb{L}$ be a field extension of $\mathbb{F}_q$, and in our finite setting we will always let $n = [\mathbb{L} : \mathbb{F}_q]$. Fix an $\mathbb{F}_q$-homomorphism $\gamma : A \to \mathbb{L}$; $\ker \gamma$ is then generated by an irreducible $\mathfrak{p} \in A$ of degree m. We let $\mathbb{F}_\mathfrak{p} = A/(\mathfrak{p})$ and $\mathbb{F}_\mathfrak{p}$ is isomorphic to a sub-field of $\mathbb{L}$. If $\mathbb{F}_\mathfrak{p} \cong L$, that is $n = m$, we will refer to this as the *prime-field* case. We therefore have a tower of fields

$$\mathbb{F}_q \overset{m}{\text{---}} \mathbb{F}_\mathfrak{p} \overset{n/m}{\text{---}} \mathbb{L}$$

We may refer to γ as the *characteristic map*, and $\mathfrak{p}$ as the A-*characteristic*. To understand the definitions, it may be helpful to keep in mind the analogies with elliptic curves, and to that end we have the following table illustrating the role that each object plays and the corresponding object for number fields.

Object	Definition	Number field analogue
A	Regular functions on C	$\mathbb{Z}$
$\mathbb{F}_\mathfrak{p}$	The “prime field”	$\mathbb{Z}_p$
K	Fraction field of A	$\mathbb{Q}$
K_∞	The completion of K with respect to the fixed place	$\mathbb{R}$
C_∞	The completion of the algebraic closure $\overline{K}_\infty$	$\mathbb{C}$

Table 4.1: The “dictionary” between the number field and function field cases

The latter two objects, C_∞ and K_∞ play a role in the construction of Drinfeld modules as A-lattices of C_∞ in much the same way that elliptic curves can be constructed as 2-dimensional $\mathbb{Z}$-lattices in the complex plane. This construction, however, will not be explicitly used in this work.

Example 4.1.1. *We will let $q = 5$, $n = 4$. We can concretely represent $\mathbb{L} = \mathbb{F}_{5^4} = \mathbb{F}_5[y]/(y^4 + 4y^2 + 4y + 2)$. We let $\gamma(x) = y \bmod y^4 + 4y^2 + 4y + 2$, in which case $\mathfrak{p} = x^4 + 4x^2 + 4x + 2$ and we find ourselves in the prime field case with $\mathbb{F}_\mathfrak{p} \cong \mathbb{L}$.*

We could instead take $\gamma(x) = y^3 + y^2 + y + 3 \bmod y^4 + 4y^2 + 4y + 2$, in which case $\mathfrak{p} = x^2 + 4x + 2$ and we have $m = 2$ and $\mathbb{F}_\mathfrak{p} \cong \mathbb{F}_{5^2}$

Recall that $\mathbb{L}\{\tau\}$ is the ring of skew-polynomials subject to the commutation rule $\tau\ell = \ell^q\tau$ for all $\ell \in \mathbb{L}$. It will be helpful to note that we can canonically associate to each element of $\mathbb{L}\{\tau\}$ an operator acting on $\overline{\mathbb{L}}$ as follows:

$$\left(\textstyle\sum_i a_i\tau^i\right)(\ell) = \textstyle\sum_i a_i\ell^{q^i}.$$

4.1.2 Drinfeld Modules

Definition 4.1.2. *A **Drinfeld A-module** ϕ over $(\mathbb{L}, \gamma)$ is an $\mathbb{F}_q$-algebra homomorphism $\phi : A \to \mathbb{L}\{\tau\}$ such that for all $a \in A$ there exists $n_a \in \mathbb{L}\{\tau\}\tau$ such that*

$$\phi(a) = n_a + \gamma(a)$$

A standard convention in the literature is to use γ_a, ϕ_a in place of $\gamma(a), \phi(a)$ respectively, and we will follow this convention here. In the case where $C = \mathbb{P}^1$, the Drinfeld module is determined by ϕ_x and the *rank* of the Drinfeld module is $\deg_\tau \phi_x$. Throughout this work, for computational purposes, we will primarily be concerned with Drinfeld $\mathbb{F}_q[x]$-modules. We will often specify a rank r Drinfeld $\mathbb{F}_q[x]$-module by $\Delta_r, \ldots, \Delta_1 \in \mathbb{L}$ denoting the coefficients of ϕ_x; that is:

$$\phi_x = \textstyle\sum_{i=1}^r \Delta_i\tau^i + \gamma_x.$$

Example 4.1.3. *Consider the context of example 4.1.1 with $\mathbb{F}_q = \mathbb{F}_5$, $\mathbb{L} = \mathbb{F}_5[y]/(y^4 + 4y^2 + 4y + 2)$, and $\gamma(x) = y$. We then define a rank-2 Drinfeld $\mathbb{F}_q[x]$-module by setting*

$$\phi_x = y\tau^2 + 2\tau + y$$

Recall that in the case of elliptic curves, we have a $\mathbb{Z}$-action on the points p of the curve by adding p to itself $t \in \mathbb{Z}$ times. Analogously, a Drinfeld module ϕ induces an A-action on $\overline{\mathbb{L}}$ given by, for all $\ell \in \mathbb{L}$:

$$a * \ell = \phi_a(\ell)$$

The above construction allows us to define the torsion of a Drinfeld module.

Definition 4.1.4. *For $a \in A$, the a-**torsion** of a Drinfeld $\phi[a]$ is the set:*

$$\phi[a] = \{\ell \in \overline{\mathbb{L}} \mid a * \ell = 0\}$$

If $a \neq \mathfrak{p}$. then $\phi[a]$ is a free module over $A/(a)$ of rank r [37]. Now letting $\mathfrak{l}$ be an irreducible element of A different from $\mathfrak{p}$, and we will let $A_{\mathfrak{l}}$ be the $\mathfrak{l}$-adic completions of A. We may now construct the so-called *Tate module* as an inverse limit using the torsion data at each power $\mathfrak{l}^i$.

Definition 4.1.5. *Let ϕ be a rank r Drinfeld module. Let $\mathfrak{l}$ be a prime element of A different from $\mathfrak{p}$. The $\mathfrak{l}$-adic Tate module of ϕ, $T_{\mathfrak{l}}(\phi)$ to be the $A_{\mathfrak{l}}$-module*

$$T_{\mathfrak{l}}(\phi) = \varprojlim_i \phi[\mathfrak{l}^i].$$

We will note that several authors instead take $T_{\mathfrak{l}}(\phi) = \mathrm{Hom}(K_{\mathfrak{l}}, \varprojlim_n \phi[\mathfrak{l}^n])$ where $K_{\mathfrak{l}}$ is the $\mathfrak{l}$-adic completion of K, and that these constructions are the same up to isomorphism [37]. Since each $\phi[\mathfrak{l}^i]$ is free of rank r over $A/(\mathfrak{l}^i)$, $T_{\mathfrak{l}}(\phi)$ is a free module over $A_{\mathfrak{l}}$ of rank r [20]. This construction strongly parallels the classical construction of the Tate module for abelian varieties.

4.1.3 Morphisms and Characteristic Polynomials

Definition 4.1.6. *Let $\mathbb{L}'$ be an extension of $\mathbb{L}$. An $\mathbb{L}'$-**morphism** of Drinfeld modules $u : \phi \to \psi$ is a $u \in \mathbb{L}'\{\tau\}$ such that $u\phi_a = \psi_a u$ for all $a \in A$.*

We will typically consider $\mathbb{L}$-morphisms, and the unqualified term *morphism* will refer to this case. The motivation for this definition can be more clearly understood in terms of the induced A-module on $\overline{\mathbb{L}}$, as the definition is equivalent to the requirement that the following diagram commutes.

$$\begin{tikzcd}
\overline{\mathbb{L}} \arrow[r, "\phi_a"] \arrow[d, "u"] & \overline{\mathbb{L}} \arrow[d, "u"] \\
\overline{\mathbb{L}} \arrow[r, "\psi_a"] & \overline{\mathbb{L}}
\end{tikzcd}$$

We will use $\mathrm{Hom}(\phi, \psi)$ to denote the set of morphisms from ϕ to ψ and $\mathrm{End}(\phi) = \mathrm{Hom}(\phi, \phi)$. $\mathrm{End}(\phi)$ is the centralizer for $\phi(A)$, and in particular of ϕ_x, in $\mathbb{L}\{\tau\}$. A non-zero element of $\mathrm{Hom}(\phi, \psi)$ is referred to as an *isogeny* and two Drinfeld modules ϕ, ψ are *isogenous* if there is a non-zero member of $\mathrm{Hom}(\phi, \psi)$. An *isomorphism* of Drinfeld modules is a $u \in \mathrm{Hom}(\phi, \psi) \bigcap \mathbb{L}$. Recalling $[\mathbb{L} : \mathbb{F}_q] = n$, $\mathrm{End}(\phi)$ contains the so-called *Frobenius endomorphism*, τ^n.

Since elements of $\mathrm{End}(\phi)$ commute with $\phi(A)$, each $u \in \mathrm{End}(\phi)$ induces a linear map $t(u)_a : \phi[a] \to \phi[a]$ for any $a \in A$ whose action is given by $t(u)_a(\ell) = u(\ell)$ for $\ell \in \phi[a]$. This can therefore be lifted to a linear map $t(u) \in \mathrm{End}(T_{\mathfrak{l}}(\phi))$.

Definition 4.1.7. *The* ***characteristic polynomial*** $\mathrm{CharPoly}(u)$ *of* $u \in \mathrm{End}(\phi)$ *is the characteristic polynomial of* $t(u) \in \mathrm{End}(T_{\mathfrak{l}}(\phi))$.

The minimal polynomial of u is defined in an analogous manner. Since $T_{\mathfrak{l}}(\phi)$ is free of rank r over $A_{\mathfrak{l}}$, $\mathrm{CharPoly}(u)$ has degree r, and its coefficients lie in the canonical inclusion of A in $A_{\mathfrak{l}}$ [33, Cor. 3.4]. Moreover, the coefficients of the characteristic polynomial do not depend on the choice of $\mathfrak{l}$ [33, Cor. 3.4], and when the Drinfeld module map is applied to the coefficients of $\mathrm{CharPoly}(u)$, it annihilates u as a skew-polynomial [34]. To put this more concretely, the characteristic polynomial has the form

$$\mathrm{CharPoly}(u) = X^r + a_{r-1}X^{r-1} + \ldots + a_iX^i + \ldots + a_1X + a_0$$

with each $a_i \in A$ such that

$$u^r + \phi_{a_{r-1}} u^{r-1} + \ldots + \phi_{a_i} u^i + \ldots + \phi_{a_1} u + \phi_{a_0} = 0. \tag{4.1}$$

For Drinfeld $\mathbb{F}_q[x]$-modules, the coefficients satisfy a degree bound

$$\deg(a_i) \leq \tfrac{d(r-i)}{r}$$

which can be viewed as an analogue of the Hasse bound of theorem 3.2.1. The minimal polynomial $\text{MinPoly}(u) = \sum_{i=0}^{r_1} b_i x^i \in A[x]$ of u is also the polynomial of minimal degree r_1 such that

$$u^{r_1} + \phi_{b_{r_1-1}} u^{r_1-1} + \ldots + \phi_{b_i} u^i + \ldots + \phi_{b_1} u + \phi_{b_0} = 0.$$

Moreover, we have the following lemma.

Lemma 4.1.8. *[2, 58] Let $u \in \text{End}(\phi)$. Then*

$$(\text{MinPoly}(u))^{r_2} = \text{CharPoly}(u)$$

with r_2 such that $r_2 | \frac{n}{m}$.

This divisibility requirement immediately leads to the following corollary.

Corollary 4.1.9. *If $n = m$ or $\gcd(n/m, r) = 1$, then $\text{MinPoly}(u) = \text{CharPoly}(u)$ for all $u \in \text{End}(\phi)$.*

Example 4.1.10. *Consider the context of example 4.1.3 with $\mathbb{F}_q = \mathbb{F}_5$, $\mathbb{L} = \mathbb{F}_5[y]/(y^4 + 4y^2 + 4y + 2)$, $\gamma(x) = y$, and a rank-2 Drinfeld $\mathbb{F}_q[x]$-module given by*

$$\phi_x = y\tau^2 + 2\tau + y$$

Its characteristic polynomial is

$$X^2 + (2x^2 + 3x + 1)X + 3x^4 + 2x^2 + 2x + 1$$

We can verify that

$$\tau^{2n} + \phi_{2x^2+3x+1}\tau^n + \phi_{3x^4+2x^2+2x+1} = 0$$

For the remainder of this work, the characteristic polynomial of a Drinfeld module refers to the characteristic polynomial of the Frobenius endomorphism τ^n. Two Drinfeld modules are isogenous if and only if their characteristic polynomials coincide [2]. In the case of Drinfeld modules over $\mathbb{F}_q[x]$, the term a_0 is known as the *Frobenius norm*, and can be computed directly via the following formula:

$$a_0 = (-1)^{r+n(r+1)} N_{\mathbb{F}_{\mathfrak{p}}/\mathbb{F}_q}(\Delta_r)^{-1}\mathfrak{p}^{\frac{n}{m}}. \tag{4.2}$$

4.2 Elementary Algorithms on Drinfeld Modules

In this section, we will focus on the basic algorithms associated with Drinfeld modules over $\mathbb{F}_q[x]$.

4.2.1 Computing Images and Inversions of the Drinfeld Map

Perhaps the first algorithmic question to come to mind with regards to Drinfeld modules is the complexity of computing the skew polynomial ϕ_a given a polynomial $a = \sum_{i=0}^{d} a_i x^i \in \mathbb{F}_q[x]$. For the inversion operation, we are given a skew polynomial $s \in \mathbb{L}\{\tau\}$ with $\deg s = rd$, and wish to either return a such that $s = \phi_a$ or determine that $s \notin \phi(\mathbb{F}_q[x])$.

Proposition 4.2.1. *Let ϕ be a Drinfeld $\mathbb{F}_q[x]$-module of rank r and $a \in \mathbb{F}_q[x]$ be a polynomial of degree d. There exist deterministic algorithms to compute $\phi_a \in \mathbb{L}\{\tau\}$ with costs*

- *$(d^2nr\log^2 q)^{1+o(1)}$ bit operations*
- *$O(\mathrm{SM}(rd,n,q))$ bit operations if* SM *is super-linear in the degree parameter.*

Conversely, given a skew polynomial s of degree rd there exists an algorithm to compute $a \in \mathbb{F}_q[x]$ such that $s = \phi_a$ or determines that $s \notin \phi(\mathbb{F}_q[x])$ with a bit cost of $(d^2nr\log^2 q)^{1+o(1)}$.

The first approach computes powers ϕ_{x^i} for $i \leq d = \deg a$, from which we can reconstruct $\phi_a = \sum_{i=0}^{d} a_i\phi_{x^i}$ at a bit cost of $(d^2nr\log q)^{1+o(1)}$. To compute the required skew polynomials, observe that since

$$\phi_{x^{i+1}} = \phi_x\phi_{x^i} = \left(\sum_{i=1}^{r}\Delta_i\tau^i + \gamma(x)\right)\phi_{x^i}$$

the coefficients $f_{i,j}$ therefore satisfy the recurrence:

$$f_{i+1,j} = \gamma(x)f_{i,j} + \sum_{k=1}^{r}\Delta_i f_{i,j-k}^{q^k} \tag{4.3}$$

With the usual pre-processing step for computing polynomial representations of Frobenius powers, this recurrence allows the computation of $\phi_{x^2},\ldots,\phi_{x^d}$ in $(d^2nr\log^2 q)^{1+o(1)}$ bit operations.

We also present an alternative algorithm for computing ϕ_a based on a classic divide-and-conquer approach [30]. First, observe that a key bottleneck of the previous method was the insistence on computing $\phi_{x^2},\ldots,\phi_{x^d}$. If we instead wish to compute ϕ_{x^d} for a single choice of d, we can use a repeated squaring procedure based on skew-polynomial multiplication costing $\mathrm{SM}(rd,n,q)$. Write $a = a_0 + x^{\lfloor d/2\rfloor}a_1$, with $\deg a_0, \deg a_1 \leq \lfloor d/2\rfloor + 1$. The cost of computing ϕ_a then reduces to the cost of computing ϕ_{a_0}, ϕ_{a_1}, and $\phi_{x^{\lfloor d/2\rfloor}}$ as well as computing the product $\phi_{x^{\lfloor d/2\rfloor}}\phi_{a_1}$. The latter two operations cost $\mathrm{SM}(rd,n,q)$; from this we can conclude that the complexity $\mathrm{Im}(d)$ of computing the image of a degree d element of $\mathbb{F}_q[x]$ satisfies the following recurrence:

$$\mathrm{Im}(d) = 2\mathrm{Im}(d/2) + O(\mathrm{SM}(rd, n, q)).$$

Since SM is super-linear in the degree parameter, the cost of computing ϕ_a is at most $O(\mathrm{SM}(rd, n, q))$, which completes the complexity analysis of computing ϕ_a.

Now turning to the question of inversion, this problem can be reduced to solving a linear system over $\mathbb{L}$ which we will construct below. First, let:

$$\begin{aligned}
a &= \sum_{i=0}^{d} a_i x^i \\
s &= \sum_{j=0}^{rd} s_j \tau^j \\
\phi_a &= \sum_{i=0}^{d} a_i \sum_{j=0}^{ri} f_{i,j}\tau^j = \sum_{j=0}^{rd} \left(\sum_{i=\lfloor j/r \rfloor}^{d} a_i f_{i,j} \right) \tau^j.
\end{aligned}$$

This gives $rd+1$ equations, with coefficients $f_{i,j} \in \mathbb{L}$, in $d+1$ unknowns. By extracting equations corresponding to every r^{th} coefficient of s, we obtain the upper-triangular system

$$\begin{bmatrix} f_{0,0} & f_{1,0} & \dots & f_{d,0} \\ 0 & f_{1,r} & \dots & f_{d,r} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & f_{d,rd} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_d \end{bmatrix} = \begin{bmatrix} s_0 \\ s_r \\ \vdots \\ s_{rd} \end{bmatrix}.$$

The diagonal entries $f_{i,ri}$ of this system are the leading coefficients of ϕ_{x^i}, which are powers of Δ_r, and therefore non-zero. This implies that the system has a unique solution, which we may solve for using at most $O(d^2)$ operations in $\mathbb{L}$. A valid solution a to the inversion problem exists if and only if the values $a_i \in \mathbb{F}_q$ for all $1 \leq i \leq d$. Including the cost of computing the required entries $f_{i,j}$ gives a total bit complexity of $(d^2 nr \log^2 q)^{1+o(1)}$.

4.2.2 Computing Field Actions

We recall that each ϕ_a acts as a field endomorphism on $\mathbb{L}$, and more generally any field extension contained in $\overline{\mathbb{L}}$. For $a \in \mathbb{F}_q[x]$ of degree d and for $\alpha \in \mathbb{L}$, our goal is to determine $\phi_a(\alpha)$. The direct approach leverages the algorithms of §4.2.1 to compute the skew polynomial ϕ_a. Using modular composition, we can compute $\sigma^i(\alpha)$ for $i \leq dr$ at a total bit cost of $(drn \log q)^{1+o(1)}$, leaving the dominant cost coming from the computation of ϕ_a which can be done at a cost of either $(d^2 nr \log^2 q)^{1+o(1)}$, if we wish to recover the intermediate powers of ϕ_x, or $\mathrm{SM}(rd, n, q))$ otherwise. If we wish to compute the evaluation at a set $\{\alpha_1, \ldots, \alpha_s\}$ instead, we can use the multipoint evaluation algorithms of §2.4.3 or §7.1. This leaves us with the following corollary.

Corollary 4.2.2. *Let ϕ be a Drinfeld $\mathbb{F}_q[x]$-module of rank r, $a \in \mathbb{F}_q[x]$, and let $\{\alpha_1, \ldots, \alpha_s\} \subset \mathbb{L}$. There exists an algorithm to compute $\{\phi_a(\alpha_1), \ldots, \phi_a(\alpha_s)\}$ with a bit complexity of $\tilde{O}(\mathrm{SM}(\max(s,d), n, q) + \max(s,d)^{\log_2(3)} n \log q + T^*)$ where $T^* = \mathrm{SM}(rd, n, q))$ or $T^* = (d^2 nr \log^2 q)^{1+o(1)}$.*

For a fixed linear form $\ell \in \mathbb{L}^*$, $\ell : \mathbb{L} \to \mathbb{F}_q$, we can invoke the transposition principle [30] to conclude that computing the linear mapping $\ell(\phi_a)$ given by $\ell(\phi_a)(\alpha) = \ell(\phi_a(\alpha))$ can be done at the same asymptotic cost as computing the field action $\phi_a(\alpha)$.

4.3 The j-Invariant

The classical j-invariant for elliptic curves is notable for classifying isomorphism classes of curves. We will now state the notion of j-invariant for Drinfeld modules:

Definition 4.3.1. *[63, Def 2.1] Let ϕ be a Drinfeld module with $\phi_x = \sum_{i=0}^{r} \Delta_i \tau^i$. Let $h \leq r-1$ and $k_1, \ldots, k_h$ be integers with $1 \leq k_1 < \ldots < k_h \leq r-1$ and let $\delta_1, \ldots, \delta_h$ be integers and $\delta_r > 0$ a positive integer satisfying*

$$\sum_{i=1}^{h} \delta_i (q^{k_i} - 1) = \delta_r (q^r - 1). \tag{4.4}$$

*Then the $J^{\delta_1,\ldots,\delta_h}_{k_1,\ldots,k_h}$ **j-invariant** is defined as:*

$$J^{\delta_1,\ldots,\delta_h}_{k_1,\ldots,k_h}(\phi) = \frac{\Delta_{k_1}^{\delta_1} \cdots \Delta_{k_h}^{\delta_h}}{\Delta_r^{\delta_r}}.$$

*A $J^{\delta_1,\ldots,\delta_h}_{k_1,\ldots,k_h}$ j-invariant is **basic** if, in addition to the previous requirements, it satisfies:*

- $\delta_i \leq \frac{q^r-1}{q^{\gcd(i,r)}-1}$
- $\gcd(\delta_1, \ldots, \delta_h, \delta_r) = 1$

For Drinfeld modules of rank 2, there is a single j-invariant given by $J_1^{q+1} = \frac{\Delta_1^{q+1}}{\Delta_2}$ which is also basic. The basic j-invariants classify Drinfeld modules up to isomorphism, in a direct analogy of the classical j-invariant of elliptic curves.

Proposition 4.3.2. *[63, Thm 2.2] Let ϕ, ψ be rank r Drinfeld A-modules over $(\mathbb{L}, \gamma)$. Then ϕ and ψ are $\overline{\mathbb{L}}$-isomorphic if and only if $J^{\delta_1,\ldots,\delta_h}_{k_1,\ldots,k_h}(\phi) = J^{\delta_1,\ldots,\delta_h}_{k_1,\ldots,k_h}(\psi)$, for all basic j-invariants $J^{\delta_1,\ldots,\delta_h}_{k_1,\ldots,k_h}$.*

Finding all possible basic j-invariants for Drinfeld modules over $\mathbb{L}$ requires the determination of all integer solutions of equation (4.4) satisfying the size constraints $\delta_i \leq \frac{q^r-1}{q^{\gcd(i,r)}-1}$. Algorithms to find such solutions based on finding integral points on polyhedra exist, but are exponential in the number parameters $h \leq r$. As we will see later, computing the space of morphisms between two Drinfeld modules can be done by computing the kernel of a linear system. This allows us to determine whether two Drinfeld modules are isomorphic in polynomial time, making the j-invariant approach obsolete in practice.

4.4 Computing the Characteristic Polynomial

The question of computing the characteristic polynomial, particularly in the rank 2 case and particularly of the Frobenius endomorphism, has seen some prior study in the literature [34] [29] [56] [15]. We will cover the most significant approaches preceding our work in the next sections, and the complexity results are summarized in the following theorem.

Theorem 4.4.1. *Let ϕ be a Drinfeld module. There exists deterministic algorithms to compute the characteristic polynomial of the Frobenius endomorphism τ^n with complexities*

1.1 $(n^{\omega+1}r^{\omega}\log q + n^3 r\log^2 q)^{1+o(1)}$ *when* $r = 2$ *or* $\mathrm{CharPoly}(\tau^n) = \mathrm{MinPoly}(\tau^n)$

1.2 $(n^3 r\log^2 q)^{1+o(1)}$ *in the prime field case* $\mathbb{L} - \mathbb{F}_{\mathfrak{p}}$*.*

Furthermore, there exist Monte Carlo randomized algorithms to compute the characteristic polynomial when $r = 2$ *with complexities*

2.1 $(n^2\log q\log n)^{1+o(1)}$ *when* $q > \frac{n}{2}$

2.2 $(n^2\log^2 q)^{1+o(1)}$

4.4.1 The Direct Approach

We may exploit the fact that an endomorphism is annihilated by its characteristic polynomial to construct a linear system as done by Gekeler in [34]. This algorithm is applicable if $r = 2$ or if $\mathrm{CharPoly}(\tau^n) = \mathrm{MinPoly}(\tau^n)$. Define

$$a_i = \sum_{j=0}^{n(r-i)/r} a_{i,j}x^j$$

$$\phi_{x^j} = \sum_{k=0}^{rj} f_{j,k}\tau^k$$

Then we have

$$\tau^{nr} + \sum_{i=1}^{r-1}\sum_{j=0}^{\frac{n(r-i)}{r}}\sum_{k=0}^{n(r-i)} a_{i,j}f_{j,k}\tau^{k+ni} + \phi_{a_0} = 0 \tag{4.5}$$

After computing the coefficients of ϕ_{x^j} up to $j \le \frac{n(r-1)}{r}$, equating coefficients of each term τ^i from the left hand side of equation (4.5) to 0, we obtain a system with nr equations with $\sum_{i=1}^{r-1} \frac{ni}{r} = \frac{n(r-1)}{2}$ unknowns. The coefficients $f_{j,k}$ are extracted from the computation $\phi_{x^2}, \dots, \phi_{x^k}$ using the algorithm of proposition 4.2.1 in $(k^2 nr \log^2 q)^{1+o(1)}$ bit operations. Computing all coefficients of the linear system takes $(n^3 r \log^2 q)^{1+o(1)}$ bit operations, and solving it costs $(n^{\omega+1} r^{\omega} \log q)^{1+o(1)}$.

For the preceding algorithm to return a solution, we require that there is a unique solution to the system extracted from equation (4.5); that is, there is a unique monic polynomial of degree r that annihilates τ^n with constant coefficient a_0. When $r = 2$, either $\mathrm{MinPoly}(\tau^n) = \mathrm{CharPoly}(\tau^n)$ or $\mathrm{MinPoly}(\tau^n) = x - \sqrt{a_0}$ and $\mathrm{CharPoly}(\tau^n) = (x - \sqrt{a_0})^2$. Since any solution of (4.5) must have $\mathrm{MinPoly}(\tau^n)$ as a factor, we conclude $\mathrm{CharPoly}(\tau^n)$ is the unique solution to equation (4.5) with constant term a_0.

When $r > 2$, the system has a unique solution if and only if $\mathrm{CharPoly}(\tau^n) = \mathrm{MinPoly}(\tau^n)$. Recalling that $\mathrm{CharPoly}(\tau^n) = \mathrm{MinPoly}(\tau^n)^{r_2}$ for some exponent r_2, if $r_2 > 1$ and $r > 2$ then $\deg \mathrm{CharPoly}(\tau^n) - \deg \mathrm{MinPoly}(\tau^n) = r_3 > 1$. In this case, we can multiply $\mathrm{MinPoly}(\tau^n)$ by any monic polynomial of degree r_3, whose constant term can be suitably chosen to satisfy the constraints of equation (4.5), to obtain an annihilating polynomial of τ^n. This completes the analysis of item 1.1 of theorem 4.4.1.

4.4.2 An Algorithm for the Prime Field Case

When $\mathbb{L} = \mathbb{F}_{\mathfrak{p}}$, a simpler algorithm for computing the characteristic polynomial due to Garai and Papikian [29] can be used.

Define

$$\overline{g}_k = \sum_{i=k}^{r} \phi_{a_i} \tau^{ni} \quad \text{and} \quad g_k = \sum_{i=0}^{k-1} \phi_{a_i} \tau^{ni}.$$

Then we must have that the lowest degree term of $\overline{g}_k$ is $\gamma(a_k)\tau^{nk}$. Since $g_k + \overline{g}_k = 0$, the coefficient of τ^{nk} in g_k is $-\gamma(a_k)$. This permits the computation of all the γ_{a_k} as follows:

- Compute a_0 using equation (4.2), which has the form $a_0 = (-1)^{r+n(r+1)} N_{\mathbb{L}/\mathbb{F}_q}(\Delta_r)^{-1}\mathfrak{p}$.

- Pre-compute $\phi_x, \ldots . \phi_{n(r-1)/r}$.
- For each $1 \leq k < r$, compute g_k as above and set $-\gamma_{a_k}$ equal to the coefficient of τ^{nk}.

Since each coefficient a_k satisfies $\deg a_k \leq n$, we can recover the original polynomial in $\mathbb{F}_q[x]$ from its image under γ. This algorithm has a cost of $(n^3 r \log^2 q)^{1+o(1)}$ arising from the computation of the terms $\phi_x, \ldots . \phi_{n(r-1)/r}$. This completes the analysis of item 1.2 of theorem 4.4.1.

4.4.3 A Schoof-like Approach when $r = 2$

Here we will present an algorithm that originally appeared in my MMath thesis in [55]. For this exposition we will assume that $q > \frac{n}{2}$ and $r = 2$. Choose $e_0, \ldots, e_{n/2} \in \mathbb{F}_q$ and reduce the equation $\tau^{2n} - \phi_{a_1}\tau^n + \phi_{a_0} = 0$ modulo ϕ_{x-e_i} to obtain

$$\tau^{2n} \bmod \phi_{x-e_i} - a_1(e_i)\tau^n \bmod \phi_{x-e_i} + a_0(e_i) = 0 \tag{4.6}$$

By computing $\tau^n \bmod \phi_{x-e_i}$, we obtain $a_1(e_i)$, and doing this for all $\frac{n}{2} + 1$ evaluation points e_i and interpolating allows us to recover a_1. We proceed to compute $\tau^n \bmod \phi_{x-e_i}$ as follows: let

$$\tau^j = \nu_j + \mu_j \tau \bmod \phi_x - e_i$$

Then we have

$$\tau^{j+1} = \nu_j^q \tau + \mu_j^2 \tau^2 = \nu_j^q \tau + \mu_j^q \big(-\frac{\gamma_x - e_i}{\Delta_2} - \frac{\Delta_1}{\Delta_2}\tau \big)$$

from which we obtain the recurrence relations $\nu_{j+1} = -\frac{\gamma_x - e_i}{\Delta_2}\mu_j^q$ and $\mu_{j+1} = \nu_j^q - \frac{\Delta_1}{\Delta_2}\mu_j^q$. Now setting $\alpha = -\frac{\gamma_x - e_i}{\Delta_2}$, $\beta = -\frac{\Delta_1}{\Delta_2}$, we define

$$T = \begin{bmatrix} 0 & \alpha \\ 1 & \beta \end{bmatrix}.$$

The desired coefficients ν_n, μ_n are then given by the matrix product

$$\begin{bmatrix} \nu_n \\ \mu_n \end{bmatrix} = TT^{[1]} \dots T^{[n-1]} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Recall that $a^{[i]} = a^{q^i}$ for $a \in \mathbb{L}$. For matrices T, we will let $T^{[i]}$ denote entrywise exponentiation by q^i. Now defining $P_j = TT^{[1]} \dots T^{[j]}$, we have a relation

$$P_{2j+1} = P_j P_j^{[j+1]}. \tag{4.7}$$

After the usual pre-processing step for computing Frobenius powers, this recurrence allows us to compute the desired coefficients ν_n, μ_n at a cost of $(n \log q \log n)^{1+o(1)}$. Once these have been determined, we can substitute the coefficients into equation (4.6), where we have

$$a_1(e_i)(\nu_n + \mu_n \tau) = (\nu_n + \mu_n \tau)^2 + a_0(e_i) \bmod \phi_x - e_i. \tag{4.8}$$

We can expand this to obtain

$$a_1(e_i)(\nu_n + \mu_n \tau) = \left(\nu_n \mu_n + \nu_n^q \mu_n - \frac{\Delta_1}{\Delta_2} \mu_n \mu_n^q \right) \tau + \nu_n^2 - \frac{\gamma_x - e_i}{\Delta_2} \mu_n^2 + a_0(e_i) \bmod \phi_x - e_i. \tag{4.9}$$

From the preceding equations, we can conclude that, if $\mu_n \neq 0$, then we can determine the evaluation of the trace at e_i without pre-computing the norm using the expression:

$$a_1(e_i) = \nu_n + \nu_n^q - \frac{\Delta_1}{\Delta_2} \mu_n^q.$$

Otherwise, we obtain

$$a_1(e_i) = \nu_n + \frac{a_0(e_i)}{\nu_n} - \frac{(\gamma_x - e_i)\mu_n^2}{\Delta_2 \nu_n}.$$

The dominant step in this computation is therefore calculation of ν_n, μ_n for each of at least $\frac{n}{2}$ values of e_i, which has an overall bit cost contribution of $(n^2 \log q \log n)^{1+o(1)}$, giving the complexity required for item 2.1 of Theorem 4.4.1.

4.4.4 Using Hankel Systems

Here we will present another algorithm for rank 2 Drinfeld modules ϕ given in [55]; a Monte Carlo randomized approach inspired by ideas due to Shoup [68] as well as Wiedemann's algorithm for solving linear systems [75]. The aim of this approach is to construct a structured *Hankel* system that can be solved for the coefficients of a_1.

Definition 4.4.2. *A $t \times t$ matrix is Hankel if it is of the form*

$$\begin{bmatrix} a_1 & a_2 & a_3 & \dots & a_t \\ a_2 & a_3 & a_4 & \dots & a_{t+1} \\ a_3 & a_4 & a_5 & \dots & a_{t+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_t & a_{t+1} & a_{t+2} & \dots & a_{2t-1} \end{bmatrix}$$

We can solve linear systems defined by Hankel matrices over a field at a cost of $O(\mathrm{Mul}(n,q)\log(n))$ field operations [9]. The following lemma, due to Kaltofen and Pan, will play a crucial role in determining when Hankel systems have a unique solution.

Lemma 4.4.3. *[45, Lemma 1] Consider a linear sequence $\{\ell_i\}_{i=0}^{\infty}$ over a field $\mathbb{F}$, and let d denote the degree of the sequence's minimal polynomial. For any $t > 0$ let T_t be a matrix of the form:*

$$\begin{bmatrix} \ell_0 & \ell_1 & \dots & \ell_{t-1} \\ \ell_1 & \ell_2 & \dots & \ell_t \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{t-1} & \ell_t & \dots & \ell_{2t-2} \end{bmatrix}$$

Then $\det T_d \neq 0$ and for any $m > d$, $\det T_m = 0$.

Lemma 4.4.4. *Let $\Gamma \in \mathbb{F}_q[x]$ denote the minimal polynomial of ϕ_x acting on $\mathbb{L}$ and let κ be its degree. Then $\kappa \geq \frac{n}{2}$.*

Proof. Substituting $\phi_x = \Delta_2\tau^2 + \Delta_1\tau + \gamma_x$ into Γ, for some coefficients $s_0, \ldots, s_{2\kappa} \in \mathbb{L}$, we obtain an expression

$$\Gamma(\phi_x) = s_0 + s_1\tau + \ldots + s_{2\kappa}\tau^{2\kappa} = 0$$

Now recall the canonical identification $\tau \mapsto \sigma$ with linear operators $\sigma(s) = s^q$ introduced in section §2.4. Appealing to Artin's independence of characters, we have that the set $\{\sigma^i\}_{0\le i<n}$ satisfy no non-trivial $\mathbb{L}$-linear relation. Therefore if $\kappa < \frac{n}{2}$, we must have that all coefficients $s_i = 0$. However we also have that $s_{2\kappa}$ must be a power of Δ_2 obtained as the coefficient of $(\Delta_2\tau^2)^\kappa$, and therefore $s_{2\kappa} = \Delta_2^{(1-q^{2\kappa})/(1-q)} \neq 0$, giving a contradiction. $\square$

We now construct a linear system as follows: choose a random projection $\ell : \mathbb{L} \to \mathbb{F}_q$, and a random element $\alpha \in \mathbb{L}$. Write

$$a_1 = \sum_{i=0}^{\lfloor n/2 \rfloor} c_i x^i$$

and set $r = \alpha + \phi_{a_0}(\alpha)$. It then follows from the definition of the characteristic polynomial that, for any $j > 0$, we have

$$\sum_{i=0}^{\lfloor n/2 \rfloor} c_i \ell(\phi_{x^{i+j}}(\alpha)) = \ell(\phi_{x^j}(r)). \tag{4.10}$$

Then we have a linearly generated sequence $(\ell(\phi_x^i(\alpha)))_{i\ge 0} \subset \mathbb{F}_q$, from which we construct a Hankel system as seen in lemma (4.4.3) to solve for the coefficients c_i. Examining the lemma once again, we observe that if n is even and if $\kappa = \frac{n}{2}$, the dimension $\lfloor \frac{n}{2} \rfloor + 1$ system required to solve for all coefficients is singular. To overcome this, we can pre-compute $c_{n/2}$ using a formula due to Jung [43]

$$c_{n/2} = \mathrm{Tr}_{\mathbb{F}_{q^2}/\mathbb{F}_q}(\mathrm{N}_{\mathbb{L}/\mathbb{F}_{q^2}}(\Delta_2)^{-1}).$$

By replacing r with $\tilde{r} = \alpha + \phi_{a_0}(\alpha) - c_{n/2}\phi_{\lfloor \frac{n}{2} \rfloor}(\alpha)$ the system arising from equation (4.10) becomes

$$\begin{bmatrix} \ell(\alpha) & \dots & \ell(\phi_{x^{\lfloor \frac{n}{2} \rfloor -1}}(\alpha)) \\ \vdots & & \vdots \\ \ell(\phi_{x^{\lfloor \frac{n}{2} \rfloor -1}}(\alpha)) & \dots & \ell(\phi_{x^{2\lfloor \frac{n}{2} \rfloor -1}}(\alpha)) \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ c_{\lfloor \frac{n}{2} \rfloor -1} \end{bmatrix} = \begin{bmatrix} \ell(\tilde{r}) \\ \vdots \\ \ell(\phi_{x^{\lfloor \frac{n}{2} \rfloor -1}}(\tilde{r})) \end{bmatrix} \tag{4.11}$$

Letting $\Gamma_{\ell,\alpha}$ denote the minimal polynomial of the sequence $(\ell(\phi_x^i(\alpha)))_{i\geq 0}$. Combining lemma (4.4.3) with lemma (4.4.4), in the case where $\Gamma_{\ell,\alpha} = \Gamma$ the linear system of (4.11) has a unique solution since $\deg \Gamma_{\ell,\alpha} \geq \frac{n}{2}$. Using a result of Wiedemann [75] as well as the DeMillo-Lipton-Zippel-Schwartz lemma (which will be discussed in more detail later), we have that $\Gamma_{\ell,\alpha} = \Gamma$ with probability at least $\max(1/(12\max(1, \log_q \nu)), 1 - 2n/q)$ [45]. Consequently, the algorithm successfully returns the Frobenius trace with probability bounded by the same value. The entire algorithm then proceeds as follows:

1. Choose ℓ, α, and determine $\Gamma_{\ell,\alpha}$ using the Berlekamp-Massey algorithm. If $\deg \Gamma_{\ell,\alpha} \geq \frac{n}{2}$, proceed to step 2.
2. Compute $\ell(\phi_{x^i}(\alpha))$ for $0 \leq i \leq 2\lfloor \frac{n}{2} \rfloor - 1$ and $\ell(\phi_{x^i}(\tilde{r}))$ for $0 \leq i \leq \lfloor \frac{n}{2} \rfloor - 1$.
3. Construct and solve the linear system as given by equation (4.11).

Determining $\Gamma_{\ell,\alpha}$ costs at most $(n^2 \log q)^{1+o(1)}$ bit operations. Computing the entries $\ell(\phi_{x^i}(\alpha))$, $\ell(\phi_{x^i}(\tilde{r}))$ can be done through repeated applications of the operator ϕ_x, costing a total of $(n^2 \log^2 q)^{1+o(1)}$ bit operations. The Hankel system can be solved for a cost of $(n^2 \log q)^{1+o(1)}$ bit operations, leaving an overall bit complexity of $(n^2 \log^2 q)^{1+o(1)}$.

4.4.5 Narayanan's Algorithm

For rank 2 Drinfeld modules where the minimal polynomial of ϕ_x over $\mathbb{F}_q$ has degree n, Narayanan [59] gave a Monte Carlo algorithm. Recall that $\phi_x \in \mathbb{L}\{\tau\}$ can be interpreted as a linear operator on $\mathbb{L}/\mathbb{F}_q$.

Lemma 4.4.5. *[33] The characteristic polynomial of ϕ_x as a linear operator on $\mathbb{L}/\mathbb{F}_q$ is* $1 - a_1 + a_0$.

By choosing a random projection $\ell : \mathbb{L} \to \mathbb{F}_q$, and a random element $\alpha \in \mathbb{L}$, we again obtain a sequence $(\ell(\phi_x^i(\alpha)))_{i\geq 0} \subset \mathbb{F}_q$. As mentioned previously, the minimal polynomial of this sequence can be determined via the Berlekamp-Massey algorithm. Moreover, with probability at least $\max(1/(12\max(1, \log_q \nu)), 1 - 2n/q)$, we have that $\Gamma_{\ell,\alpha}$ coincides with the minimal polynomial of ϕ_x, from which we can deduce the Frobenius trace.

The main computational step lies in determining $2n$ terms of the sequence $(\ell(\phi_x^i(\alpha)))_{i\geq 0}$ to which we may apply the Berlekamp-Massey algorithm to compute $\Gamma_{\ell,\alpha}$. To do this, Narayanan invokes the automorphism projection algorithm of Kaltofen and Shoup [46] to obtain a complexity with order at most $n^{(\omega+1)/2}$. This, however, requires that ϕ_x is an automorphism on $\mathbb{L}$ which will not hold in general. Whether this algorithm can be adapted in a way that maintains sub-quadratic complexity overall is an open question.

4.4.6 Recent Work for Computing Norms of Isogenies and Characteristic Polynomials of Endomorphisms

Recent work designing algorithms for computing characteristic polynomials of endomorphisms of Drinfeld modules of arbitrary rank, as well as norms for more general morphisms, was done in [15]. These results were developed contemporaneously with the writing of this thesis, so while we will not discuss them in detail readers are encouraged to review the paper for further background on the subject.

Chapter 5

Cohomology of Drinfeld Modules

5.1 Derivations and De Rham Cohomology

The construction of the de Rham and crystalline cohomology modules appears to be due to Gekeler, but the sources are not easily accessible. For this reason, we will follow and cite the exposition due to Anglès in [2]. We begin by letting $N = \mathbb{L}\{\tau\}\tau$, and ϕ be a rank r Drinfeld module over $\mathbb{L}$.

Definition 5.1.1. *A* ***derivation*** *on ϕ is an $\mathbb{L}$-linear map $\eta : A \to N$ such that for all $a \in A$*

$$\eta_{ab} = \gamma_a \eta_b + \eta_a \phi_b$$

We will denote the space of all derivations $D(\phi, \mathbb{L})$. Observing that any derivation is determined entirely by its value η_x at x, the mapping $\eta \mapsto \eta_x$ gives an $\mathbb{L}$-linear bijection from $D(\phi, \mathbb{L})$ onto N. This will allow us to view the cohomology spaces as rings of skew polynomials; however the derivational construction is useful to keep in mind for the parallels with the usual construction of the de Rham cohomology for manifolds.

Let $A_{\mathbb{L}} = \mathbb{L} \otimes_{\mathbb{F}_q} A$; in the particular case where $A = \mathbb{F}_q[x]$, we have $A_{\mathbb{L}} \cong \mathbb{L}[x]$. We may define an $A_{\mathbb{L}}$-action on $D(\phi, \mathbb{L})$ given by, for any $\ell \in \mathbb{L}$, $a, b \in A$, $\eta \in D(\phi, \mathbb{L})$:

$$((\ell \otimes a) * \eta)_b = \ell \eta_b \phi_a$$

This action turns $D(\phi, \mathbb{L})$ into a left $A_{\mathbb{L}}$-module. A similar action defined by $((\ell \otimes a) * n) = \ell s \phi_a$ for all $a \in A$, $s \in N$ turns N into an $A_{\mathbb{L}}$-module, and the evaluation map $\eta \mapsto \eta_x$ gives an isomorphism $D(\phi, \mathbb{L}) \to N$ as $A_{\mathbb{L}}$-modules. We are therefore free to view $D(\phi, \mathbb{L})$ as essentially the skew polynomial subring N equipped with the previously described $A_{\mathbb{L}}$-action.

This construction is also closely related, in fact almost entirely identical, to the so-called *Anderson motive* of a Drinfeld module [1]. The Anderson motive $M(\phi)$ can be constructed by turning $\mathbb{L}\{\tau\}$ into an $A_{\mathbb{L}}$ module under the $A_{\mathbb{L}}$-action $(c \otimes a)(u) = cu\phi_a$ for $c \in \mathbb{L}$, $a \in A$, $u \in \mathbb{L}\{\tau\}$. It follows that $M(\phi) \cong N \cong D(\phi, \mathbb{L})$, and we are free to work with each construction and obtain similar results. However, to highlight the parallels with the classical case we will proceed with the crystalline cohomology constructions of Gekeler and Anglès. In particular, we have the following lemma:

Lemma 5.1.2. *[2, Lemma 2.3] $D(\phi, \mathbb{L})$ is a projective module over $A_{\mathbb{L}}$ of rank r.*

In the case where $A = \mathbb{F}_q[x]$, $D(\phi, \mathbb{L})$ is projective over the coefficient ring $\mathbb{L}[x]$ and therefore free. The $\mathbb{L}[x]$ action on $D(\phi, \mathbb{L})$ in this case can be explicitly written out as, for $a = \sum_i a_i x^i \in \mathbb{L}[x]$, $s \in N$:

$$\left(\sum_i a_i x^i \right) * s = \sum_i a_i s \phi_{x^i}.$$

For the remainder of this work, we will omit the $*$ operator indicating scalar multiplication by $\mathbb{L}[x]$.

Definition 5.1.3. *A derivation is* ***strictly inner*** *if it is of the form*

$$\eta_a = \gamma_a n - n\phi_a$$

for some $n \in N$.

Denote the set of strictly inner derivations by $D_{si}(\phi, \mathbb{L})$. It is straightforward to verify that $D_{si}(\phi, \mathbb{L})$ is an $A_{\mathbb{L}}$-submodule of $D(\phi, \mathbb{L})$. We now have sufficient background to formally define the de Rham cohomology.

Definition 5.1.4. *[2, Def. 2.1] The* ***de Rham cohomology*** *$H^*_{dR}(\phi, L)$ of a Drinfeld module ϕ is the $A_{\mathbb{L}}$-module quotient*

$$H^*_{dR}(\phi, L) = D(\phi, \mathbb{L})/D_{si}(\phi, \mathbb{L})$$

In the case $A = \mathbb{F}_q[x]$, we have a simpler description of $D_{si}(\phi, \mathbb{L})$ as $D_{si}(\phi, \mathbb{L}) = (x - \gamma_x)D(\phi, \mathbb{L})$.

5.2 Crystalline Cohomology

Let $A = \mathbb{F}_q[x]$. Now define a map $\xi : \mathbb{L}[x] \to \mathbb{L}$ given by $a \otimes \ell \mapsto \gamma(a)\ell$. Then let $I = \ker \xi = (x - \gamma_x)$ and observe that I is an ideal lying above $\mathfrak{p}$ in $A_{\mathbb{L}}$. Now set $W_k = \mathbb{L}[x]/I^k$ and define $W(\mathbb{L})$ to be the I-adic completion of $\mathbb{L}[x]$, given by the inverse limit $W(\mathbb{L}) = \varprojlim_k W_k$.

We have $W(\mathbb{L}) \cong \mathbb{L}[[x - \gamma_x]]$ and each element of $\mathbb{F}_q[x]$ naturally embeds into $W(\mathbb{L})$ via a map $\iota : \mathbb{F}_q[x] \to W(\mathbb{L})$ giving the I-adic expansion. Moreover, $W(\mathbb{L})$ comes equipped with projections $\pi_k : W(\mathbb{L}) \to W_k$. We let $\iota_k = \iota \circ \pi_k$, and since I^k lies above $\mathfrak{p}^k$, there exists a homomorphism $\chi_k : W_k \to \mathbb{F}_q[x]/\mathfrak{p}^k$ such that $\iota_k \circ \chi_k$ gives the quotient modulo $\mathfrak{p}^k$. We can represent the situation with the following commutative diagram.

$$\begin{tikzcd}
 & W(\mathbb{L}) \arrow[d, "\pi_k"] \\
\mathbb{F}_q[x] \arrow[ur, "\iota"] \arrow[r, "\iota_k"] \arrow[dr, "\theta_k : f(x) \mapsto f(z) \bmod \mathfrak{p}(z)^k"'] & W_k \arrow[d, dashed, "\chi_k"] \\
 & \mathbb{F}_q[x]/(\mathfrak{p}(x)^k)
\end{tikzcd}$$

We will now show how to compute the maps χ_k effectively. We begin with a representation $\mathbb{L} = \mathbb{F}_q[t]/(\ell(t))$, and write W_k as the bi-variate quotient ring $W_k = \mathbb{F}_q[t, z]/(\ell(t), (z-\gamma_x)^k)$. We can convert elements of $\mathbb{L}$ to their bi-variate representation $\mathbb{L} = \mathbb{F}_q[x, t]/(\mathfrak{p}(x), g(x, t))$ using modular composition to give an isomorphism

$$W_k = \mathbb{F}_q[t, z]/(\ell(t), (z - \gamma_x)^k) \to \mathbb{F}_q[x, t, z]/(\mathfrak{p}(x), g(x, t), (z - x)^k)$$

In [39, §4.5], the authors gave a so-called *tangling* algorithm for explicitly computing an isomorphism $\mathbb{F}_q[x, z]/(\mathfrak{p}(x), (z - x)^k) \to \mathbb{F}_q[x]/(\mathfrak{p}(x)^k)$ in $(km \log q)^{1+o(1)}$ bit operations. This isomorphism can be applied coefficient-wise to powers of t to obtain an isomorphism

$$\mathbb{F}_q[x, t, z]/(\mathfrak{p}(x), g(x, t), (z - x)^k) \to \mathbb{F}_q[x, t]/(\mathfrak{p}(x)^k, G_k(x, t))$$

With G_k a polynomial of t-degree $\frac{n}{m}$ chosen such that $W_k \cong \mathbb{F}_q[x, t]/(\mathfrak{p}(x)^k, G_k(x, t))$.

Definition 5.2.1. *[2, Def 2.5] The* ***Crystalline Cohomology*** *of a Drinfeld module ϕ is the $W(\mathbb{L})$-module*

$$H^*_{crys}(\phi, \mathbb{L}) = W(\mathbb{L}) \otimes_{\mathbb{L}[x]} D(\phi, \mathbb{L})$$

The ***precision-k cohomology*** *$H^*_k(\phi, \mathbb{L})$ is the W_k-module:*

$$H^*_k(\phi, \mathbb{L}) = D(\phi, \mathbb{L})/I^k\, D(\phi, \mathbb{L}) \simeq H^*_{\mathrm{crys}}(\phi, \mathbb{L})/I^k\, H^*_{\mathrm{crys}}(\phi, \mathbb{L}).$$

It follows from lemma (5.1.2) that $H^*_{crys}(\phi, \mathbb{L})$ is a free module of rank r over $W(\mathbb{L})$, and each $H^*_k(\phi, \mathbb{L})$ is free over W_k of the same rank. In particular, $H^*_{dR}(\phi, L) \cong D(\phi, \mathbb{L})/I\, D(\phi, \mathbb{L})$ and is a dimension r vector space over $A_{\mathbb{L}}/I \cong \mathbb{L}$. Each cohomology space has a canonical basis corresponding to the derivations $\eta_x = \tau^i$ for $1 \leq i \leq r$ [2].

5.3 Endomorphisms and Characteristic Polynomials

Given an endomorphism $u \in \mathrm{End}(\phi)$, we obtain a corresponding module endomorphism $\hat{u} \in \mathrm{End}(H^*_{\mathrm{crys}}(\phi, \mathbb{L}))$ given by, for all $a \in A$, $\eta \in H^*_{\mathrm{crys}}(\phi, \mathbb{L})$:

$$\hat{u}(\eta)_a = \eta_a u.$$

Since $H^*_{\text{crys}}(\phi, \mathbb{L})$ is free of rank r, we may represent each endomorphism $\hat{u}$ as an $r \times r$ matrix with entries in $W(\mathbb{L})$, and set $\text{CharPoly}(\hat{u})$ to be the formal characteristic polynomial of this matrix. We can obtain a related endomorphism $\tilde{u}$ acting on $M(\phi)$ given by $\tilde{u}(s) = su$ for $s \in M(\phi)$ (recall that $M(\phi)$ is the space of skew polynomials with a particular $A_{\mathbb{L}}$ action). The following theorem of Anglès relates $\text{CharPoly}(\hat{u})$ to $\text{CharPoly}(u)$.

Theorem 5.3.1. *[2, Thm. 3.2] For $u \in \text{End}(\phi)$ and let $\hat{u} \in \text{End}(H^*_{\text{crys}}(\phi, \mathbb{L}))$ be its induced endomorphism. Then we have that*

$$\text{CharPoly}(\hat{u}) = \text{CharPoly}(u).$$

Similar results for $M(\phi)$ were shown in [15, Thm. 2.8], and in particular we also have $\text{CharPoly}(\tilde{u}) = \text{CharPoly}(u)$ which we will reproduce the proof from [15] below.

Proof. Fix $a \in \mathbb{F}_q[x]$, and let $M(\phi)_a = M(\phi)/aM(\phi)$. Observe that $\phi[a]$ is a free module of rank r over $\mathbb{F}_q[x]/(a)$, which implies that as an $\mathbb{F}_q$-vector space we have $\dim \phi[a] = r \deg(a)$. Similarly, $M(\phi)_a$ is a free module over $\mathbb{L}[x]/(a)$ of rank r, and $\dim_{\mathbb{L}} M(\phi)_a = r \deg(a)$. We turn $\phi[a]$ into a vector space over $\overline{\mathbb{L}}$ by an extension by scalars, that is, set $\overline{\phi}[a] = \overline{\mathbb{L}} \otimes_{\mathbb{F}_q} \phi[a]$ and $\overline{M}(\phi)_a = \overline{\mathbb{L}} \otimes_{\mathbb{L}} M(\phi)_a$. Both are finite dimensional vector spaces over $\overline{\mathbb{L}}$ and since $\dim_{\mathbb{F}_q} \phi[a] = \dim_{\mathbb{L}} D(\phi, a)$, we have $\dim_{\overline{\mathbb{L}}} \overline{\phi}[a] = \dim_{\overline{\mathbb{L}}} \overline{M}(\phi)_a$.

Let $\tilde{u}_a$ denote the module endomorphism induced on $M(\phi)_a$ by $\tilde{u}$. Our first goal will be to show that $\text{CharPoly}(t(u)_a) = \text{CharPoly}(\tilde{u}^*_a)$ where $\tilde{u}^*_a$ is the dual morphism of $\tilde{u}_a$ acting on $\overline{M}(\phi)^*_a = \text{Hom}(\overline{M}(\phi)_a, \overline{\mathbb{L}})$.

Let $\theta : \overline{\phi}[a] \to \overline{M}(\phi)^*_a$ given by

$$\begin{aligned} \theta : \overline{\phi}[a] &\to \overline{D}(\phi.a)^* \\ c \otimes z &\mapsto (b \otimes s \mapsto bcs(z)) \end{aligned}$$

with $b, c \in \overline{\mathbb{L}}$, $z \in \phi[a]$, and $s \in \mathbb{L}\{\tau\}$.

Lemma 5.3.2. *The map $\theta : \overline{\phi}[a] \to \overline{M}(\phi)^*_a$ defined above is an isomorphism.*

Proof. It suffices to show it is injective. Suppose we have an element $\ell \in \overline{\phi}[a]$ such that $\theta(\ell) = 0$. Let $z_1, \ldots, z_d \in \phi[a]$ be linearly independent, such that we can write $\ell = \sum_{i=1}^{d} c_i \otimes z_i$ for $c_i \in \overline{\mathbb{L}}\backslash 0$. Among the non-zero choices for ℓ, there exists at least one choice such that the integer d is minimal and suppose ℓ is chosen such that this is the case. This further implies that the z_i are $\mathbb{F}_q$-linearly independent, since if say $z_d = \sum_{i=1}^{d-1} t_i z_i$ then we also have $\ell = \sum_{i=1}^{d-1}(c_i + t_i) \otimes z_i$. The requirement $\theta(\ell) = 0$ implies that for all $s \in M(\phi)_a$ we have

$$\sum_{i=1}^{d} c_i s(z_i) = 0. \tag{5.1}$$

In particular, to show $\ell \in \ker\theta$ it suffices to show that equation (5.1) holds when $s = \tau^j$ for all $j > 0$. Therefore, for any $j \in \mathbb{N}$ we must have

$$\sum_{i=1}^{d} c_i z_i^{q^j} = 0$$

So we must also have:

$$\sum_{i=1}^{d} c_i^q z_i^{q^{j+1}} = 0 \text{ and } \sum_{i=1}^{d} c_d^{q-1} c_i z_i^{q^{j+1}} = 0$$

We therefore obtain:

$$\sum_{i=1}^{d-1} (c_i^q - c_d^{q-1} c_i) z_i^{q^{j+1}} = 0 \tag{5.2}$$

If $c_i^q - c_d^{q-1}c_i = 0$, then $(c_i/c_d)^{q-1} = 1$ which implies $c_i/c_d \in \mathbb{F}_q$. If this where the case, then $\sum_{i=1}^{d}(c_i/c_d)z_i = 0$, which contradicts linear independence of the z_i over $\mathbb{F}_q$. It follows that equation (5.2) is a non-trivial sum; but we can also conclude from equation (5.2) that $\sum_{i=1}^{d-1}(c_i^q - c_d^{q-1}c_i) \otimes z_i^q \in \ker\theta$, which contradicts the minimality of d. $\square$

Now recall that for an endomorphism u of ϕ, the corresponding map on $\phi[a]$ induces a map on $\overline{\phi}[a]$ by setting $t(u)_a(c \otimes z) = c \otimes u(z)$. Similarly, recall that u induces an action $\tilde{u}_a$ on $M(\phi)_a$ given by right multiplication by u. The corresponding dual map $\tilde{u}_a^*$ on $\overline{M}(\phi)_a^*$ therefore acts on elements of the form $\theta(c \otimes z)$ as:

$$\hat{u}^*(\theta(c \otimes z)) = b \otimes s \mapsto bcs(u(z)) = \theta(c \otimes u(z))$$

Observe that $\tilde{u}_a^*$ and $t(u)_a$ are similar under θ, and we can conclude their characteristic polynomials coincide. Since characteristic polynomials are invariant under extensions by scalars and taking dual maps, we conclude that $\text{CharPoly}(t(u)_a) = \text{CharPoly}(\tilde{u}_a)$ for all choices of a. With respect to a fixed set of elements simultaneously forming a basis for $M(\phi)$ and $M(\phi)_a$, the matrix obtained from $\tilde{u}_a$ is simply the matrix for $\tilde{u}$ with respect to the basis with coefficients reduced modulo a. From this, we can conclude $\text{CharPoly}(\tilde{u}) = \text{CharPoly}(u)$ as desired.

$\square$

Chapter 6

Algorithms for Computing the Characteristic Polynomials of Endomorphisms of Drinfeld Modules

In this chapter, we will present new algorithms, as well as extensions of previous algorithms. We will primarily focus on the computation of the characteristic polynomial of the Frobenius endomorphism τ^n. Moreover, the algorithms presented here will focus on improving asymptotic complexity in the parameter $n = [\mathbb{L} : \mathbb{F}_q]$. This is due to one of the original motivations for studying the computation of the characteristic polynomial in the rank 2 case: it is a subroutine in algorithms for polynomial factorization over finite fields [59], for which its complexity in n is a bottleneck.

6.1 Schoof-Like Algorithms

6.1.1 The Rank 2 Case

One major assumption of the algorithm of §4.4.3 was that the base field $\mathbb{F}_q$ contained enough elements, at least $\frac{n}{2}+1$, to allow us to interpolate a_1 from its evaluation at elements of $\mathbb{F}_q$. We will now extend this algorithm to remove the requirement $q \geq \frac{n}{2}+1$. This result is based on work published in [56].

Theorem 6.1.1. *There exists a deterministic algorithm to compute the characteristic polynomial of the Frobenius endomorphism of a rank 2 Drinfeld module ϕ with a bit complexity of $(n^2 \log q + n \log^2 q)^{1+o(1)}$.*

Proof. After computing a_0 explicitly using equation (4.2), we need only compute a polynomial $a_1 \in \mathbb{F}_q[x]$ with $\deg a_1 \leq \frac{n}{2}$ such that

$$\tau^{2n} - \phi_{a_1}\tau^n + \phi_{a_0} = 0$$

For any $E \in \mathbb{F}_q[x]$ with $\deg E < d$ for some choice of d, we reduce equation right-modulo ϕ_E to obtain

$$\tau^n \phi_{a_1 \bmod E} \bmod \phi_E = \tau^{2n} + \phi_{a_0 \bmod E} \bmod \phi_E. \tag{6.1}$$

Now consider the $\mathbb{F}_q$-linear mapping $T : \mathbb{L}\{\tau\}_{2d} \to \mathbb{L}\{\tau\}_{2d}$ given by $S \mapsto \tau S \bmod \phi_E$ for any $S \in \mathbb{L}\{\tau\}_{2d}$. Write

$$\phi_E = \sum_{i=0}^{2d} s_i \tau^i$$
$$\tau^i \bmod \phi_E = \sum_{j=0}^{2d-1} \mu_{i,j}\tau^j$$
$$\phi_{x^i} = \sum_{j=0}^{2i} f_{i,j}\tau^j.$$

Now expand the action of T on τ^i:

$$\tau^{i+1} = \sum_{j=0}^{2d-1} \mu_{i,j}^q \tau^{j+1} = \sum_{j=1}^{2d-1} \mu_{i,j-1}^q \tau^j + \sum_{j=0}^{2d-1} -\frac{s_j}{s_{2d}} \mu_{i,2d-1}^q \tau^j.$$

Recall that for a (skew) polynomial, the (transposed) companion matrix $\mathcal{C}_{\phi_E}$ is given by

$$\mathcal{C}_{\phi_E} = \begin{bmatrix} -\frac{s_{2d-1}}{s_{2d}} & -\frac{s_{2d-2}}{s_{2d}} & \dots & -\frac{s_1}{s_{2d}} & -\frac{s_0}{s_{2d}} \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}$$

Let $\mathfrak{C} = \mathcal{C}_{\phi_E}^T$. Let $\hat{s}$ denotes the standard d dimensional $\mathbb{L}$-vector representing $s \in \mathbb{L}\{\tau\}/\phi_E$. In particular, we have:

$$\hat{\tau}^i = \begin{bmatrix} \mu_{i,2d-1} \\ \vdots \\ \mu_{i,0} \end{bmatrix}$$

then the action of $T(\hat{s})$ is given by

$$T\hat{s} = \mathfrak{C}(\hat{s})^{[1]}.$$

We can invert this action to obtain the inverse operator

$$T^{-1}\hat{s} = \left(\mathfrak{C}^{-1}\hat{\tau}^i\right)^{[-1]}.$$

Using the same procedure based on equation (4.7), we can explicitly compute $T^n(\hat{s})$ and $T^{-n}(\hat{s})$ at a cost of $(d^\omega n \log q)^{1+o(1)}$ using the following relations

$$T^n\hat{s} = \mathfrak{C}\mathfrak{C}^{[1]} \dots \mathfrak{C}^{[n-2]}\mathfrak{C}^{[n-1]}\hat{s}$$
$$T^{-n}\hat{s} = \mathfrak{C}^{[-1]}\mathfrak{C}^{[-2]} \dots \mathfrak{C}^{[1-n]}\mathfrak{C}\hat{s}.$$

Having computed a_0 using equation (4.2), we can compute ϕ_E and $\phi_{a_0 \bmod E}$ at a bit cost of $(d^2 n \log^2 q)^{1+o(1)}$ using the algorithm of proposition 4.2.1. Let $a_1 \bmod E = \sum_{i=0}^{d-1} \alpha_i x^i$, and expand equation (6.1) to obtain:

$$\tau^n \sum_{\substack{i<d \\ j \leq 2(d-1)}} \alpha_i f_{i,j} \tau^j = \tau^{2n} + \phi_{a_0 \bmod E} \tag{6.2}$$

Which we can rewrite as a system over $\mathbb{L}$ of the form

$$\begin{bmatrix} f_{0,0} & f_{1,0} & \dots & f_{d,0} \\ 0 & f_{1,2} & \dots & f_{d,2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & f_{d-1,2(d-1)} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{d-1} \end{bmatrix} = T^n \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} + T^{-n}(\hat{\phi}_{a_0 \bmod E}) \tag{6.3}$$

Note that on the left hand side, we have the same matrix that is obtained when attempting to compute inverse images of the Drinfeld map as seen in section §4.2.1. This system can be solved, uniquely, at a bit cost of $(d^2 n \log q)^{1+o(1)}$. We have an overall cost of $(d^\omega n \log q)^{1+o(1)}$ from computing $a_1 \bmod E$.

To reconstruct a_1, we need to compute $a_1 \bmod E_i$ for sufficiently many distinct irreducible polynomials $\{E_1, E_2, \dots, E_w\} \subset \mathbb{F}_q[x]$ such that $\deg E_i \leq d$ and $\deg \prod_{i=1}^{w} E_i > \frac{n}{2}$. Here we will recall a classical result telling us that the product of all irreducible of degree dividing d is $x^{q^d} - x$ seen in, for example, [32, Fact 3.3]. This implies it suffices to select the E_i from polynomials of degree at most $d = \lceil \log_q(n+1) \rceil$ of which there are at most $q^{d+1} \leq q^{2\log_q(n)} = n^2$. Testing each such polynomial for irreducibility has a cost polynomial in $d \in O(\log(n))$ as well as $\log q$ using the algorithms of [32, Thm. 7.5], incurring an overall cost of $(n^2 \log q)^{1+o(1)}$ to generate the E_i.

Computing $a_1 \bmod E_i$ for $O(n/\log n)$ choices of E_i yields a total cost of $(n^2 \log q)^{1+o(1)}$. Reconstructing a_1 from the polynomials $a_1 \bmod E_i$ for all i costs $O(n^2 \log q)$ bit operations. Adding in the usual cost of $n \log^2 q$ for pre-computing polynomial representations of Frobenius powers, we obtain the complexity of theorem 6.1.1. □

6.1.2 The Schoof-like Algorithm in any Rank

In this section, we will present a generalization of the algorithm seen in section (4.4.3) to arbitrary rank, but keep the requirement of a large base field $\mathbb{F}_q$, requiring that $q \gg nr$. As before let $\mathbb{F}_q$ be a field of order q and $\mathbb{L}$ a degree n extension of $\mathbb{F}_q$. Let ϕ be an $\mathbb{F}_q[x]$-Drinfeld module over $\mathbb{L}$ of rank r. Note that Drinfeld modules of rank r are parametrized over $\mathbb{L}^{r+1}$ via the correspondence $(\Delta_0, \ldots, \Delta_r) \mapsto (\phi_x = \Delta_r\tau^r + \ldots + \Delta_1\tau + \Delta_0)$. We will note that in this case that the characteristic map γ is determined by Δ_0 and will not be fixed. We can expand this to a parametrization over $\mathbb{F}_q^{n(r+1)}$ by introducing parameters $\alpha_{i,j}$ such that $\Delta_i = \sum_j \alpha_{i,j}\lambda_j$ for a basis $\{\lambda_j\}_{j=1}^n$ of $\mathbb{L}/\mathbb{F}_q$.

As in the rank-two case, the goal will be to evaluate CharPoly(τ^n) at τ^n and reduce right modulo ϕ_{x-e} for at least $N = \lfloor\frac{n(r-1)}{r}\rfloor + 1$ choices of $e \in \mathbb{F}_q$. The solution of this system yields the evaluations $\{a_i(e)\}_{i=1}^{r-1}$, from which we can interpolate to recover the entire characteristic polynomial. Our algorithm will therefore require that we randomly choose a set of evaluation points $E = (e_1, \ldots, e_N) \in \mathbb{F}_q^N$. It will turn out to be the case that for not all choices of ϕ_x and evaluation points e, that the resulting system reduced modulo ϕ_{x-e} can be solved uniquely to guarantee the correct result is returned.

We also note that the algorithm will only return a correct result if CharPoly(τ^n) = MinPoly(τ^n); if this is not the case, the linear system we obtain can not be solved uniquely, which is a failure mode that is already detected by the algorithm.

Theorem 6.1.2. *Let ϕ be a rank r Drinfeld module over $\mathbb{L}$ such that* CharPoly(τ^n) = MinPoly(τ^n). *There exists an algorithm such that if* $\gcd(n, r) = 1$ *or* $\gcd(n, r-1) = 1$, *and $q > 2nr$, then the algorithm computes the characteristic polynomial of the Frobenius endomorphism of ϕ if the following conditions hold.*

1. *ϕ_x lies outside of a hypersurface in $\mathbb{F}_q^{n(r+1)}$ defined by a polynomial $\mathfrak{f}$ of degree at most $(n-1)(r-1)$.*

2. *A randomly chosen vector $E = (e_1, \ldots, e_n) \in \mathbb{F}_q^N$ lies outside of a hypersurface in $\mathbb{F}_q^N$ of degree at most $n^2r + \frac{n^2}{2}$.*

3. *A randomly chosen projection $\ell : \mathbb{L} \to \mathbb{F}_q$ lies outside of a hypersurface in $\mathbb{F}_q^n$ of degree at most $n(r-1)$.*

Moreover, this algorithm runs with a bit complexity of $(r^{\omega_2/2+1}n^2 \log q + n \log^2 q)^{1+o(1)}$.

We will devote the rest of this section to the proof of the preceding theorem. As usual, we will fix a Drinfeld module ϕ defined by $\phi_x = \Delta_r \tau^r + \ldots + \Delta_1 \tau + \Delta_0$. As with the rank 2 case, we will begin by reducing equation (4.1) modulo a skew polynomial of the form ϕ_{x-e} for $e \in \mathbb{F}_q$.

$$\tau^{nr} + a_{r-1}(e)\tau^{n(r-1)} + \ldots + a_1(e)\tau^n + a_0(e) = 0 \mod \phi_{x-e}. \tag{6.4}$$

By solving equation (6.4) for the values $\{a_i(e)\}_{i=0}^{r-1}$, for at least n distinct choices of e, we can interpolate to recover the coefficients of the characteristic polynomial. As an aside, we observe that this method will only be viable for Drinfeld modules ϕ where $\mathrm{CharPoly}(\tau^n) = \mathrm{MinPoly}(\tau^n)$. As done previously, for a fixed $\psi \in \mathbb{L}\{\tau\}$ we will let $\hat{s}$ denote the $(\deg \psi)$-dimensional coefficient vector of $s \bmod \psi$ with entries in $\mathbb{L}$ for all $s \in \mathbb{L}\{\tau\}$; the ψ will usually be specified in context so we will omit this dependency. Moreover, let

$$R(\psi) = \begin{bmatrix} \hat{\tau}^{n(r-1)} & \ldots & \hat{\tau}^n & \hat{1} \end{bmatrix} \in M_{r \times r}(\mathbb{L}).$$

Let $R_e = R(\phi_{x-e})$. Equation (6.4) therefore leads to a linear system that can be written in the following form:

$$R_e \begin{bmatrix} a_{r-1}(e) \\ \vdots \\ a_1(e) \\ a_0(e) \end{bmatrix} = \hat{\tau}^{nr}. \tag{6.5}$$

Equation (6.5) can be solved if and only if $\mathrm{rank}(R_e) \geq r$ i.e. when the set $\{\hat{\tau}^{nj}\}_{0 \leq j < r}$ is linearly independent. While this won't hold in general, we intend to show that the number of choices of Drinfeld module ϕ and evaluation point e_i for which this fails is small when

$q \gg nr$. To do this, we can make use of a classical result in probabilistic polynomial identity testing, the DeMillo-Lipton-Schwartz-Zippel lemma, which we will introduce here, for a more precise probability analysis.

Lemma 6.1.3. *[66, DeMillo-Lipton-Schwartz-Zippel lemma] Let $\mathbb{F}$ be a field and $S \subset \mathbb{F}$. Let $f \neq 0$ be a multivariate polynomial $f(x_1, \ldots, x_t) \in \mathbb{F}_q[x_1, \ldots, x_t]$ of total degree $d > 0$. For $c_1, \ldots, c_t \in S$ chosen uniformly at random from S we have that $\Pr[f(c_1, \ldots, c_t) = 0] \leq \frac{d}{|S|}$.*

Rather than provide the probability analysis explicitly, we will show that our algorithm succeeds when ϕ, as well as the chosen evaluations points $e_1, \ldots, e_N$ and projection ℓ, exist outside of hypersurfaces whose degrees are determined by polynomials in n, r. The first step in the proof of theorem 6.1.2 will be to establish the existence of a polynomial $\mathfrak{f}$ acting on a parametrization of skew polynomials $\psi \in \mathbb{L}\{\tau\}$ such that if $\mathrm{rank}(R(\psi)) < r$ then $\mathfrak{f}(\psi) = 0$.

We will suppose that the set $\{\hat{\tau}^{nj} \bmod \psi\}_{j \leq r}$ satisfies an $\mathbb{L}$-relation. Then there exist $\beta_i \in \mathbb{L}$ and $P \in \mathbb{L}\{\tau\}$ such that:

$$\sum_{i=1}^{r} \beta_i \tau^{ni} = P\psi. \tag{6.6}$$

Write:

$$\psi = \sum_{i=0}^{r} \psi_i \tau^i$$
$$P = \sum_{i=n}^{(n-1)r} P_i \tau^i.$$

Now, recasting equation (6.6) as a linear system, set:

$$\tilde{B}(\psi) = \begin{bmatrix} \psi_r^{q^{(n-1)r}} & 0 & 0 & 0 & 0 & \dots & 0 & \dots & 0 \\ \psi_{r-1}^{q^{(n-1)r}} & \psi_r^{q^{(n-1)r-1}} & 0 & 0 & 0 & \dots & 0 & \dots & 0 \\ \psi_{r-2}^{q^{(n-1)r}} & \psi_{r-1}^{q^{(n-1)r-1}} & \psi_r^{q^{(n-1)r-2}} & 0 & 0 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \psi_0^{q^{(n-1)r}} & \psi_1^{q^{(n-1)r-1}} & \dots & \psi_r^{q^{(n-1)r-r}} & 0 & \dots & 0 & \dots & 0 \\ 0 & \psi_0^{q^{(n-1)r-1}} & \psi_1^{q^{(n-1)r-2}} & \dots & \psi_r^{q^{(n-1)r-r-1}} & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & \dots & \psi_0^{q^n} \end{bmatrix} \tag{6.7}$$

$\tilde{B}(\psi)$ is an $(nr-n+1) \times (n-1)(r-1)$ matrix where $\tilde{B}(\psi)_{i,j} = \psi_{r+j-i}^{q^{(n-1)r-j}}$ for $0 \le j \le i$ and $i \le r+j$. This gives the system

$$\tilde{B}(\psi) \begin{bmatrix} P_{(n-1)r} \\ \vdots \\ P_n \end{bmatrix} = \begin{bmatrix} \beta_r \\ 0 \\ \vdots \\ 0 \\ \beta_i \\ 0 \\ \vdots \\ 0 \\ \beta_1 \end{bmatrix} \tag{6.8}$$

Removing the rows of $\tilde{B}(\psi)$ corresponding to the positions of $\beta_1, \dots, \beta_r$, which correspond to indices $\{ni\}_{i=0}^{r-1}$ (our matrices are 0-indexed), we obtain a matrix $B(\psi)$ such that

$$B(\psi) \begin{bmatrix} P_{(n-1)r} \\ \vdots \\ P_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{6.9}$$

Then a non-zero P satisfying equation (6.6) exists only if $\det B(\psi) = 0$. Fix a basis $\{\lambda_1, \dots, \lambda_n\}$ for $\mathbb{L}/\mathbb{F}_q$; each ψ_i can be parametrized by variables $\alpha_{i,j}, e \in \mathbb{F}_q$ for $1 \le$

$i \leq n$ and $0 \leq j \leq r$ such that $\psi_i = \sum_{j=1}^n \alpha_{i,j}\lambda_j$ for $i \neq 0$ and $\psi_0 = \sum_{j=1}^n \alpha_{i,j}\lambda_j - e$ (we can view this setup as effectively parametrizing pairs (ϕ_x, e) corresponding to skew polynomials ϕ_{x-e}). Noting that $\psi_i^{q^k} = \sum_{j=1}^n \alpha_{i,j}\lambda_j^{q^k}$, we have that all entries of $B(\psi)$ are degree-1 polynomials in the variables $\alpha_{i,j}$ taking values in $\mathbb{F}_q$. Then $\mathfrak{f}(\psi, e) = \det B(\psi) \in \mathbb{F}_q[\{\alpha_{i,j}\}_{0\leq i\leq r}^{1\leq j\leq n}, e]$ with total degree at most $(n-1)(r-1)$. Suppose the partial evaluation of $\mathfrak{f}(\phi_x, e)$ at ϕ, corresponding to a fixed choice of the $\alpha_{i,j}$, does not yield the zero polynomial in e. If $\mathfrak{f}$ is not the zero polynomial, this happens outside of a hypersurface containing choices of $\phi_x \subset \mathbb{F}_q^{n(r+1)}$ defined by the simultaneous vanishing of the coefficients of $\mathfrak{f}(\phi_x, e)$ in the variables $\alpha_{i,j}$, each of which have degree at most $(n-1)(r-1)$. We recall this is part 1 of the hypotheses of Theorem 6.1.2. Under this assumption, the choices of evaluation set $E = (e_1, \dots, e_N) \subset \mathbb{F}_q^N$ such that we can not solve equation (6.5) for all e_i are defined by the vanishing of the following polynomial:

$$\prod_{i=1}^{N} \mathfrak{f}(\phi_x, e_i) \prod_{i<j} (e_i - e_j).$$

Noting this polynomial has degree at most $n^2 r + \frac{n^2}{2}$, we have now established that the algorithm will succeed outside of choices of ϕ_x and E satisfying the hypotheses of Theorem 6.1.2.

To improve the complexity of the algorithm, we will attempt to leverage the fact that the solutions to our linear system $a_0(e), \dots, a_{r-1}(e)$ lie entirely in $\mathbb{F}_q$. Select a random projection $\ell : \mathbb{L} \to \mathbb{F}_q$, and apply this projection entrywise to R_e to obtain R_e^ℓ. By representing elements of $\mathbb{L}$ as univariate polynomials $\mathbb{L} = \mathbb{F}_q[x]/(f)$ we can set $\ell = (1, \ell_1, \dots, \ell_{n-1}) \in \mathbb{F}_q^n$ such that for $\alpha = \sum_{i=0}^{n-1} \alpha_i x^i \in \mathbb{L}$ we have $\ell\big(\sum_{i=0}^{n-1} \alpha_i x^i\big) = \alpha_0 + \sum_{i=1}^{n-1} \ell_i \alpha_i$. Each entry of R_e^ℓ is then a degree-1 polynomial in the projection parameters $\ell_1, \dots, \ell_{n-1}$, and so $\det R_e^\ell$ is a degree r polynomial in the variables $\{\ell_i\}$. Observe that if $\det R_e \neq 0$ then $\det R_e^\ell$ can't be identically the zero polynomial since we can set $\ell_i = x^i \in \mathbb{L}$, under which we have $\det R_e = \det R_e^\ell \neq 0$. Given a choice of E satisfying hypothesis 2, the projected systems $R_{e_i}^\ell$ can be solved as long as our choice of ℓ avoids the hypersurface defined by the following polynomial:

$$\prod_{i=1}^{N} \det R_{e_i}^{\ell}.$$

which has degree $Nr = n(r-1)$.

It now remains to determine choices of parameters n and r such that $\mathfrak{f} = \det B(\psi)$ is not identically the zero polynomial. This can be done by showing the existence of at least one skew polynomial Ψ such that $\det B(\Psi) \neq 0$.

Lemma 6.1.4. *Let* $\det B(\psi)$ *be defined as above. If* $\gcd(n,r) = 1$*, there exists a skew-polynomial* Ψ *such that* $\det B(\Psi) \neq 0$.

Proof. Let $\Psi = \tau^r + 1$. $\tilde{B}(\Psi)$ has the form

$$\tilde{B}(\Psi) = \begin{bmatrix} 1 & 0 & 0 & \dots & \dots & \dots & \dots & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & \dots & \dots & \dots & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & \dots & \dots & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \\ 1 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots & \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & \dots & 1 & \end{bmatrix}$$

Let b_i denote the i^{th} row of $\tilde{B}(\Psi)$, and note that the rightmost 1 is contained in column i, and the second 1 if it occurs, appears at index $i - r$. Since any column contains exactly 2 non-zero entries, any non-trivial linear dependence relation between rows containing b_i must therefore contain both b_{i+r} and b_{i-r} when possible. From this it follows that, if there exists a non-trivial linear relation containing a row b_i, with $i \bmod r = d$, then there must exist a non-trivial linear dependence relation of the form

$$\sum_{\substack{0 \le i < nr-n+1 \\ i \bmod r = d}} c_i b_i = 0 \tag{6.10}$$

with all $c_i \neq 0$. Recall that the procedure for constructing $B(\overline{\psi})$ from $\tilde{B}(\overline{\psi})$ involved removing rows at indices $\{ni\}_{i=0}^{r-1}$. It follows from the hypothesis $\gcd(n, r) = 1$ that for each residue class d least one index with $i_0 \bmod r = d$ does not appear as a row of $B(\psi)$ so no relation of the form in equation (6.10) can be constructed from the rows of $B(\psi)$, which implies it has full rank and thus proving the claim. □

Lemma 6.1.5. *Let $B(\psi)$ be defined as above. If $\gcd(n, r-1) = 1$, there exists a skew-polynomial Ψ such that $\det B(\Psi) \neq 0$.*

Proof. We repeat the argument of lemma 6.1.4 with $\Psi = \tau^r + \tau$ with residues taken modulo $r-1$. One minor observation is that $\tilde{B}(\Psi)$ contains a single zero-row at index $n(r-1)$, which does not appear in $B(\Psi)$. □

Now recall that once R_e is known to have full rank, and the system of equation (6.5) can be solved for the coefficients of the characteristic polynomial. We can now summarize the algorithm of theorem 6.1.2.

1. Compute $\tau^{nk} \mod \phi_{x-e_j}$ for $\lfloor \frac{r-1}{r} n \rfloor + 1$ distinct $e_j \in \mathbb{F}_q$ and $1 \leq k \leq r$.

2. Construct the matrices R_{e_j}. For each e_i choose a random projection $\ell_j : \mathbb{L} \to \mathbb{F}_q$ and apply it to the entries of R_{e_j} to obtain $R_{e_j}^{\ell_j}$ and solve the system

$$R_{e_j}^{\ell_j} \begin{bmatrix} a_{r-1}(e) \\ \vdots \\ a_1(e) \\ a_0(e) \end{bmatrix} = \hat{\tau}^{nr} \tag{6.11}$$

   over $\mathbb{F}_q$ to obtain $a_1(e_i), \ldots, a_{r-1}(e_j)$. Repeat for n choices of e_j over $\mathbb{F}_q$.

3. Interpolate $\{a_i(e_j)\}_{j=1}^n$ for each a_j and return $X^r + \sum_{i=0}^{r-1} a_i X^i$.

Step 3 incurs a cost of $(rn \log q)^{1+o(1)}$ owing to the interpolation of r polynomials of degree at most n. The cost of constructing R_{e_i} is largely derived from computing skew polynomials $\tau^{nk} \mod \phi_{x-e_i}$. This computation can be done for each $1 \leq k \leq r$ and

$1 \leq j \leq n$ through repeated skew polynomial multiplication and division, at a total cost of $(rn\mathrm{SM}(r,n,q))^{1+o(1)}$. Computing $R^{\ell}_{e_i}$ from R_{e_i} costs $r^2 n \log q$ bit operations, and solving the linear system of equation (6.11) for an individual e_i costs at most $r^{\omega} \log q$, with a total contribution of $(r^{\omega} n \log q)^{1+o(1)}$ coming from step 2. The overall computational cost of the entire algorithm is dominated by the term $(rn\mathrm{SM}(r,n,q) + r^{\omega} n \log q)^{1+o(1)}$. Using the Puchinger and Wachter-Zeh algorithm for skew polynomial multiplication, which is the preferred approach when $r < n$, yields a complexity of $(r^{\omega_2/2+1} n^2 \log q + r^{\omega} n \log q)^{1+o(1)}$ as desired.

This completes the proof of theorem 6.1.2. The key feature of this algorithm is its quadratic complexity in n, and the removal of the restriction on the rank of the Drinfeld module. However, it is constrained primarily by the requirement of a large base field relative to nr.

6.2 A Modification of Narayanan's Algorithm

We present a modification to the algorithm of Narayanan described in §4.4.5 based on Coppersmith's block Wiedemann algorithm, which was also published in [56].

Theorem 6.2.1. *The ϕ_x be a rank 2 Drinfeld module over $(\mathbb{L}, \gamma)$. There is a Monte Carlo algorithm to compute the characteristic polynomial of the Frobenius endomorphism of ϕ if* $\deg \mathrm{MinPoly}(\phi_x) = n$ *with a bit complexity of* $(n^{1.885} \log q + n \log^2 q)^{1+o(1)}$.

Proof. Consider ϕ_x as a linear operator $\phi_x : \mathbb{L} \to \mathbb{L}$. Recall the assumption that $\Gamma = \mathrm{MinPoly}(\phi_x) = \mathrm{CharPoly}(\phi_x)$ and that Narayanan's algorithm proceeds by selecting a random projection $\ell : \mathbb{L} \to \mathbb{F}_q$ and element $a \in \mathbb{L}$ and seeks to compute the minimal polynomial of the sequence $s_i = \ell(\phi_x^i(\alpha))$.

Fix a parameter $d \in O(n^b)$ which we will later determine, and select d projections $\ell_1, \ldots, \ell_d$ mapping $\mathbb{L} \to \mathbb{F}_q$, and elements $\alpha_1, \ldots, \alpha_d \in \mathbb{L}$. We obtain a sequence of $d \times d$ matrices $(R_k)_{k>0}$ with each $(i,j)^{th}$ entry $R_{k,i,j} = \ell_i(\phi_x^k(\alpha_j))$. This sequence has a matrix generating function of the form $Q^{-1}P$ for matrices $Q, P \in M_{d \times d}(\mathbb{F}_q[x])$. From [47, Thm 2.12], we have that $\det Q$ divides the characteristic polynomial of ϕ_x, which by assumption is Γ. Moreover, with probability at least $\max(1/(12\max(1,\log_q \nu)), 1-2n/q)$ [45] we have that the minimal polynomial of $(R_{1,1,k})_{k\geq 0}$ coincides with Γ. Finally, since $\det Q$ annihilates $(R_{1,1,k})_{k\geq 0}$, $\det Q$ divides the minimal polynomial of that sequence, which will again, for generic choices of ℓ_1 and α_1, be equal to Γ. In this case we can conclude that $\det Q = \Gamma = \mathrm{CharPoly}(\phi_x)$.

We can compute Q using the PM basis algorithm of [36] which requires us to compute R_k for k at most $2n/d$. To compute the required matrices R_k, we will use a method adapted from algorithm "AP" of [46]. For some constant c which will also be determined, set $H = \lfloor (\frac{n}{2d})^c \rfloor$, $H' = \lceil \frac{n}{2Hd} \rceil$, and compute ϕ_x^H at a cost of $\mathrm{SM}(H,n,q)$. For each $1 \leq d$, pre-compute the linear operators ${}^*\ell_i^u = \ell_i \circ \phi_x^u$ for $u < H$; using the transposition argument of §4.2.2, this can be done at a bit complexity of $(Hdn\log q)^{1+o(1)}$. Note that each ${}^*\ell_i^u$ is represented as an n-dimensional row vector with coefficients in $\mathbb{F}_q$.

Next, we compute $\alpha_{j,v} = \phi_x^{Hv}(\alpha_j)$ for $v < H'$ and $j \leq d$; this can be done using the multi-point evaluation algorithm seen in theorem 7.1.1 at a cost of $(Hd^{\omega-2}n\log q)^{1+o(1)}$ for each $v < H'$, costing a total of $(d^{\omega-3}n^2\log q)^{1+o(1)}$ bit operations. Our final goal is to compute $\alpha_{u,i,j,v} = {}^*\ell_i^u(\alpha_{j,v}) = \ell_i(\phi_x^{u+Kv}(\alpha_j)) \in \mathbb{F}_q$ for $0 \leq u < H$, $0 \leq v < H'$ and $1 \leq i,j \leq d$. This can be done by computing the following $(Hd,n) \times (n, H'd)$ matrix product with coefficients in $\mathbb{F}_q$:

$$\begin{bmatrix} {}^*\ell_1^0 \\ \vdots \\ {}^*\ell_d^{H-1} \end{bmatrix} \begin{bmatrix} \alpha_{1,0} & \dots & \alpha_{d,H'-1} \end{bmatrix}.$$

This costs $(H^{3-\omega}dn^{\omega-1}\log q)^{1+o(1)} = (d^{1-c(3-\omega)}n^{\omega-1+c(3-\omega)}\log q)^{1+o(1)}$. Having computed the required matrices R_k, we can recover Q as stated before at a bit cost of $(d^{\omega-1}n\log q)^{1+o(1)}$. The determinant of Q can then be computed at a cost of $(d^{\omega-1}n\log q)^{1+o(1)}$ [52], from which we determine $\mathrm{CharPoly}(\phi_x)$. A result due to Gekeler tells is that $a_1 = 1 - \mathrm{CharPoly}(\phi_x) + a_0$ [33]. After computing a_0 using formula (4.2), we can recover the entire characteristic polynomial at a total cost of

$$(n^{b(\omega-2)+(1-b)c+1}\log q + n^{b(\omega-3)+2}\log q + n^{b(1-c(3-\omega))+\omega-1+c(3-\omega)}\log q)^{1+o(1)}$$

for a choice of parameters b, c. Accounting for a cost of $n \log^2 q$ for pre-computing polynomial representations of Frobenius maps and by setting $b = 0.183$, $c = 0.642$, $\omega = 2.372$ and we obtain the required complexity.

□

6.3 Using Hankel Systems

We generalize the approach of the algorithm in §4.4.4.

Theorem 6.3.1. *Let ϕ be a Drinfeld module of rank r over $(\mathbb{L}, \gamma)$. There exists a Monte Carlo algorithm to compute the characteristic polynomial of the Frobenius endomorphism when $\gcd(n, r) = 1$ with a bit complexity of $(n^3 r \log q)^{1+o(1)}$ and which returns the correct result with probability at least $\left(1 - \frac{n^2 r^2}{q}\right)$.*

Recall that the characteristic polynomial of the Frobenius endomorphism has the form $\sum_{i=0}^{r} a_i X^i$ with each $a_i \in \mathbb{F}_q[x]$. Using the fact that τ^n satisfies its characteristic polynomial, we have:

$$\sum_{i=1}^{r-1} \phi_{a_i} \tau^{ni} = \tau^{nr} - \phi_{a_0}. \tag{6.12}$$

Writing the coefficients of the characteristic polynomial in the form

$$a_i = \sum_{j=0}^{\lfloor \frac{n(r-i)}{r} \rfloor} a_{i,j} x^j \in \mathbb{F}_q[x]$$

We can rearrange equation (6.12) as

$$\sum_{j=0}^{\lfloor \frac{n(r-1)}{r} \rfloor} \sum_{i=1}^{r(1-\frac{j}{n})} a_{i,j} \phi_x^j \tau^{in} = \tau^{nr} - \phi_{a_0} \tag{6.13}$$

Let $\mathbb{M}$ be an extension of $\mathbb{L}$ of degree r, so that $[\mathbb{M} : \mathbb{F}_q] = nr$. Let $\alpha \in \mathbb{M}$ be such that $\{\alpha^{q^i}\}_{i=0}^{nr-1}$ is a normal basis for $\mathbb{M}/\mathbb{F}_q$; for convenience let $\alpha_i = \alpha^{q^i}$. Recall the we could view $\mathbb{L}\{\tau\}$ as a ring of linear operators acting on $\mathbb{L}/\mathbb{F}_q$; we can naturally extend these operators to act on $\mathbb{M}/\mathbb{F}_q$. For a positive integer u we can define Hankel matrices $H_u \in M_{r\times\lfloor r(1-u/n)\rfloor}(\mathbb{L}\{\tau\})$ with coefficients in $\mathbb{L}\{\tau\}$ corresponding to terms in equation (6.13) with $j = u$ with additional rows corresponding to multiplication by τ^n an additional $r-1$ times:

$$H_u = \begin{bmatrix} \phi_x^u \tau^n & \phi_x^u \tau^{2n} & \dots & \phi_x^u \tau^{\lfloor r(1-u/n)\rfloor n} \\ \phi_x^u \tau^{2n} & \phi_x^u \tau^{3n} & \dots & \phi_x^u \tau^{(1+\lfloor r(1-u/n)\rfloor)n} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_x^u \tau^{rn} & \phi_x^u \tau^{(r+1)n} & \dots & \phi_x^u \tau^{(r+\lfloor r(1-u/n)\rfloor)n} \end{bmatrix}.$$

With vectors $G \in M_{r\times 1}(\mathbb{L}\{\tau\})$ corresponding to the right-hand side:

$$G = \begin{bmatrix} \tau^{nr} - \phi_{a_0} \\ \tau^{n(r+1)} - \phi_{a_0}\tau^n \\ \vdots \\ \tau^{n(2r-1)} - \phi_{a_0}\tau^{n(r-1)} \end{bmatrix}.$$

We can then construct a block Hankel system of the form $\mathcal{H}_k$ as follows:

$$\mathcal{H} = \overbrace{\begin{bmatrix} H_0 & H_1 & \dots & H_{\lfloor n(r-1)/r\rfloor} \\ \vdots & \vdots & \ddots & \vdots \\ H_0 & H_1 & \dots & H_{\lfloor n(r-1)/r\rfloor} \end{bmatrix}}^{\lfloor (n+\frac{1}{2})(r-1)\rfloor \text{ columns}} \left.\vphantom{\begin{bmatrix} a \\ b \\ c \end{bmatrix}}\right\} n \text{ blocks}$$

$$\mathcal{G} = \begin{bmatrix} G \\ \vdots \\ G \end{bmatrix} \left.\vphantom{\begin{bmatrix} a \\ b \\ c \end{bmatrix}}\right\} n \text{ blocks}$$

Let $\alpha \in \mathbb{M}$ be a normal element for $\mathbb{M}/\mathbb{F}_q$, let $\alpha_i = \alpha^{q^i}$, and define:

$$H_u(\alpha) = \begin{bmatrix} \phi_x^u \tau^n(\alpha) & \phi_x^u \tau^{2n}(\alpha) & \dots & \phi_x^u \tau^{n(r-1-\lfloor (ur/n)\rfloor)}(\alpha) \\ \phi_x^u \tau^{2n}(\alpha) & \phi_x^u \tau^{3n}(\alpha) & \dots & \phi_x^u \tau^{(1+r-1-\lfloor (ur/n)\rfloor)n}(\alpha) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_x^u \tau^{rn}(\alpha) & \phi_x^u \tau^{(r+1)n}(\alpha) & \dots & \phi_x^u \tau^{(2r-1-\lfloor (ur/n)\rfloor)n}(\alpha) \end{bmatrix}$$

$$G(\alpha) = \begin{bmatrix} \tau^{nr}(\alpha) - \phi_{a_0}(\alpha) \\ \tau^{n(r+1)}(\alpha) - \phi_{a_0}\tau^{n}(\alpha) \\ \vdots \\ \tau^{n(2r-1)}(\alpha) - \phi_{a_0}\tau^{n(r-1)}(\alpha) \end{bmatrix}$$

Then for $\alpha_0, \alpha_r \ldots, \alpha_{(n-1)r} \in \mathbb{M}$ we can define:

$$\mathcal{H}(\alpha) = \begin{bmatrix} H_0(\alpha_0) & H_1(\alpha_0) & \ldots & H_{\lfloor n(r-1)/r \rfloor}(\alpha_0) \\ \vdots & \vdots & \ddots & \vdots \\ H_0(\alpha_{(n-1)r}) & H_1(\alpha_{(n-1)r}) & \ldots & H_{\lfloor n(r-1)/r \rfloor}(\alpha_{(n-1)r}) \end{bmatrix}$$

$$\mathcal{G} = \begin{bmatrix} G(\alpha_0) \\ \vdots \\ G(\alpha_{(n-1)r}) \end{bmatrix}$$

Finally yielding the following linear system:

$$\mathcal{H}(\alpha) \begin{bmatrix} a_{1,0} \\ a_{2,0} \\ \vdots \\ a_{1,1} \\ a_{2,1} \\ \vdots \\ a_{1,\lfloor n(r-1)/r \rfloor} \end{bmatrix} = \mathcal{G}(\alpha). \tag{6.14}$$

The rows of $\mathcal{H}$ have entries consisting of evaluations of operators of the form $\phi^u\tau^{ni}$ for $0 \le u \le \lfloor \frac{r-1}{r} n \rfloor$ and $1 \le i \le r$, with each row being evaluated at a distinct element of the normal basis generated by α. Our aim now will be to determine the conditions under which the matrix $\mathcal{H}(\alpha)$ is invertible, allowing us to solve equation (6.14) uniquely for the coefficients of the characteristic polynomial. For that, we will first define what we mean by *linear independence* of field homomorphisms.

Definition 6.3.2. *Let E, F be fields and let $\sigma_1, \ldots, \sigma_m$ be field homomorphisms $\sigma_i : E \to F$. We say that $\sigma_1, \ldots, \sigma_m$ are linearly independent if for any $\theta_1, \ldots, \theta_m \in F$ we have that if*

$$\theta_1\sigma_1(\alpha) + \ldots + \theta_m\sigma_m(\alpha) = 0$$

for all $\alpha \in F$ then we must have $\theta_1 = \ldots = \theta_m = 0$.

That is, $\sigma_1, \ldots, \sigma_m \in \mathrm{Hom}(E, F)$ are linearly independent if they are linearly independent in the usual sense when $\mathrm{Hom}(E, F)$ is viewed as an F-vector space in the natural way. We can now state and prove the following lemma.

Lemma 6.3.3. *Let E, F be a finite fields extensions of a common field K, and fix a basis $\alpha_1, \ldots, \alpha_m$ for E over K. Let $\sigma_1, \ldots, \sigma_\ell$ be linearly independent field homomorphisms $E \to F$ fixing K. Then the following matrix has a trivial right kernel:*

$$S = \begin{bmatrix} \sigma_1(\alpha_1) & \ldots & \sigma_\ell(\alpha_1) \\ \vdots & & \vdots \\ \sigma_1(\alpha_m) & \ldots & \sigma_\ell(\alpha_m) \end{bmatrix} \in M_{m\times\ell}(F).$$

Proof. Suppose S has non-trivial kernel. Then there exist $\lambda_1, \ldots, \lambda_\ell \in F$ not all 0 such that

$$\begin{bmatrix} \sigma_1(\alpha_1) & \ldots & \sigma_\ell(\alpha_1) \\ \vdots & & \vdots \\ \sigma_1(\alpha_m) & \ldots & \sigma_\ell(\alpha_m) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Then for any row-wise re-scaling with $\theta_1, \ldots, \theta_m \in K$ we must also have

$$\begin{bmatrix} \theta_1\sigma_1(\alpha_1) & \ldots & \theta_1\sigma_m(\alpha_1) \\ \vdots & & \vdots \\ \theta_m\sigma_1(\alpha_m) & \ldots & \theta_m\sigma_m(\alpha_m) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

We multiply out the left-hand side and sum the resulting entries in each row to obtain

$$\sum_{i=1}^{m}\sum_{j=1}^{m}\theta_i\lambda_j\sigma_j(\alpha_i) = \sum_{j=1}^{m}\lambda_j\sigma_j\left(\sum_{i=1}^{m}\theta_i\alpha_i\right) = 0.$$

We observe that the above relation holds for any $\theta_1, \ldots, \theta_m$, implying that $\sum_{j=1}^{m}\lambda_j\sigma_j$ is the zero morphism. Since we assumed not all the λ_i are 0, this violates the linear independence of $\sigma_1, \ldots, \sigma_m$. We conclude S has trivial kernel. $\square$

We can directly apply lemma 6.3.3 to conclude that if the set of linear operators $\{\phi_x^u\tau^{in}\}_{0\leq u<\lfloor\frac{r-1}{r}n\rfloor}^{1\leq i\leq r}$ on $\mathbb{M}$ are linearly independent, the linear system of equation 6.14 has a unique solution which must necessarily be the coefficients of the characteristic polynomial. The next phase is to establish $\mathbb{M}$-linear independence of the $\mathbb{F}_q$-linear operators $\phi_x^u\tau^{vn} : \mathbb{M} \to \mathbb{M}$. This can be done when $\gcd(n, r) = 1$ by appealing to Artin's famed *independence of characters*, which we will now state.

Lemma 6.3.4 (*Artin's Independence of Characters for Fields*)**.** *Let E, F be fields and let $\sigma_1, \ldots, \sigma_m$ be distinct field homomorphisms from E to F. Then $\sigma_1, \ldots, \sigma_m$ are linearly independent.*

Artin's independence of characters in fact holds in far greater generality, when E is replaced by a monoid and the maps σ_i are instead monoid homomorphisms $\sigma_i : E \to F^\times$ where $F^\times$ denotes the multiplicative group of the field F. Lemma 6.3.4 implies that to show a set of field homomorphisms are linearly independent, it suffices to show that they are distinct.

Lemma 6.3.5. *Let* $\gcd(n, r) = 1$*. The operators* $\{\phi_x^u\tau^{in}\}_{0\leq u<\lfloor\frac{r-1}{r}n\rfloor}^{1\leq i\leq r}$ *mapping* $\mathbb{M} \to \mathbb{M}$ *are linearly independent.*

Proof. We will show the statement for the set $S = \{\phi_x^u\tau^{in}\}_{0\leq u<\lfloor\frac{r-1}{r}n\rfloor}^{0\leq i\leq r-1}$, from which the stated lemma follows. First, we show that no two elements of $\{\phi_x^u\tau^{in}\}_{0\leq u<\lfloor\frac{r-1}{r}n\rfloor}^{0\leq i\leq r-1}$ have the same τ-degree reduced modulo rn. Suppose to the contrary that this were the case; that is:

$$ur + in \bmod nr = (\deg\phi_x^u\tau^{in}) \bmod nr = (\deg\phi_x^v\tau^{jn}) \bmod nr = vr + jn \bmod nr.$$

It then follows that

$$ur + in \bmod n = ur \bmod n = vr + jn \bmod n = vr \bmod n$$

By the assumption $\gcd(n, r) = 1$, we can divide by $r \bmod n$ and conclude $u = v$. So we must have:

$$(i - j)n = 0 \bmod nr$$

Since $|i - j| < r$, we can conclude that $i = j$, from which our claim follows. If necessary, we can perform Euclidean division of $\phi_x^u \tau^{in}$ by τ^{nr} to obtain a skew polynomial of degree at most $nr - 1$ while preserving the action on $\mathbb{M}$. Following this reduction, each element of $\phi_x^u \tau^{in} \in S$ is equivalent to an operator whose degree is $ur + in \bmod nr$, and we conclude that each element of S has a distinct reduced degree.

To complete the proof, suppose there is a linear dependence relation among the elements of S, say:

$$\sum_{u,i} \beta_{u,i} \phi_x^u \tau^{in} = 0$$

Then amongst the parameters u, i such that $\beta_{u,i} \neq 0$ there is an element with the unique highest degree mod nr, say $d_0 < nr$. This implies that τ^{d_0} as an operator can be written as a linear combination of lower order terms. However, the linear operators $\mathrm{id}, \tau, \ldots, \tau^{nr-1}$ are distinct on $\mathbb{M}$ and therefore linearly independent by Artin's lemma, leading to a contradiction. □

Using the combination of lemma 6.3.3 and lemma 6.3.5, we conclude that the matrix $\mathcal{H}(\alpha)$ is invertible when $\gcd(n, r) = 1$. We can describe the entire algorithm as follows:

1. Fix a degree r extension $\mathbb{M}$ of $\mathbb{L}$ and compute a normal basis $\{\alpha_j\}_{j<2nr}$.

2. Compute $\phi_x^u(\alpha_j)$ for $j < n$, $u < n$.

3. Compute $\phi_x^u \tau^{in}(\alpha_j)$ for $i < r$, $j < n$, $u < n$. This can be done by applying τ^{in} to $\phi_x^u(\alpha_j)$ since the operators commute.

4. Construct $\mathcal{H}(\alpha), \mathcal{G}(\alpha)$ as defined in equation (6.14). Apply a random linear projection $\ell : \mathbb{M} \to \mathbb{F}_q$ entrywise to both $\mathcal{H}(\alpha)$ and $\mathcal{G}(\alpha)$, and solve the resulting system over $\mathbb{F}_q$.

Pre-computing a normal basis for $\mathbb{M}/\mathbb{F}_q$ has a cost of $((nr)^{\omega_2/2}\log q + (nr\log^2 q))^{1+o(1)}$. Step 2 incurs a cost of $(n^3 r\log q)^{1+o(1)}$ as a result of performing $O(n^2)$ applications of ϕ_x to elements of $\mathbb{M}$, with each application costing $(rn\log q)^{1+o(1)}$ bit operations.

The resulting linear system is made up of Hankel blocks of uneven size, with $\lfloor n(r-1)/r\rfloor$ blocks spanning the columns of $\mathcal{H}$ and n blocks spanning the rows. The blocks in the i^{th} column have r rows and $r - 1 - \lfloor \frac{ir}{n}\rfloor$ columns, with total dimension of $\mathcal{H}$ system being at most $nr \times nr$. Such "Hankel-like" systems were studied in [16, Prop. 7], and the authors gave a Monte Carlo algorithm to solve such a linear system costing $O((r+n)^{\omega-1}\mathrm{Mul}(\max(n,r))\log^2(\max(n,r))$ operations in $\mathbb{M}$, and which succeeds with probability at least $\frac{1}{2}$. By applying a randomly selected projection $\ell : \mathbb{M} \to \mathbb{F}_q$ By first selecting a random projection $\ell : \mathbb{L} \to \mathbb{F}_q$ and applying it to the coefficients of $\mathcal{H}$ and $\mathcal{G}$ to generate a new system $\mathcal{H}^\ell, \mathcal{G}^\ell$ with coefficients in $\mathbb{F}_q$. Using a similar analysis to the one seen in §6.1.2, if cH is invertible, then with probability at least $1 - \frac{r^2n^2}{q}$, we have that $\mathcal{H}^\ell$ is also invertible. The cost of applying the projections entrywise is $n^2r^2\log q$ bit operations. The overall bit complexity of solving this Hankel-like system over $\mathbb{F}_q$ is then $(n^\omega r\log q)^{1+o(1)}$ using the algorithm of [16, Prop. 7].

The overall complexity of the algorithm is therefore $(n^3 r\log q)^{1+o(1)}$ owing to the cost of step 2. We observe that the requirement to compute the values $\ell(\phi_x^u\tau^{in}(\alpha_j))$ for $i < r$, $j < n$, $u < n$ bears a resemblance to Narayanan's approach in §4.4.5, and it seems likely that the techniques of §6.2 can be adapted to this setting.

6.4 Computing Characteristic Polynomials using Crystalline Cohomology

Here we will present approaches to computing the characteristic polynomial of *any* endomorphism of a Drinfeld module ϕ based on the crystalline cohomology. These algorithms were first described in [57]. Recall that by theorem (5.3.1), the characteristic polynomials $\mathrm{CharPoly}(\hat{u}) = \mathrm{CharPoly}(u)^{\iota}$, where $\hat{u}$ is the endomorphism induced on $H^*_{\mathrm{crys}}(\phi, \mathbb{L})$ by $u \in \mathrm{End}(\phi)$; for the remainder of this section we will conflate $\hat{u}$ with the corresponding skew polynomial u. Since $H^*_{\mathrm{crys}}(\phi, \mathbb{L})$ is free of rank r over $W(\mathbb{L})$, we may attempt to compute a matrix for the action of $\hat{u}$ on $H^*_{\mathrm{crys}}(\phi, \mathbb{L})$ up to precision k sufficient to recover the coefficients of $\mathrm{CharPoly}(u)$. Our objective for the rest of this section is to prove the correctness and complexity results of theorem 6.4.1.

This includes two distinct approaches for computing the characteristic polynomial of arbitrary endomorphisms, along with a specialization of the approach using the recurrence method for the special case where u is the Frobenius endomorphism. These algorithms can be viewed as a Drinfeld analog of Kedlaya's algorithm discussed in §3.4.

Theorem 6.4.1. *Let ϕ be a rank r Drinfeld $\mathbb{F}_q[x]$-module over $(\mathbb{L}, \gamma)$, and let u be any endomorphism of ϕ of degree d. Then there are deterministic algorithms to compute* $\mathrm{CharPoly}(u)$ *with the following complexities*

1. $\left(\frac{\min(dr^2, (d+r)r^{\omega-1})}{m}(d+m)n\log q + r^{\lambda}n(d+m)/m\log q + n\log^2 q\right)^{1+o(1)}$
2. $(r\mathrm{SM}(d+r, n, q) + r^{\lambda}n(d+m)/m\log q + n\log^2 q)^{1+o(1)}$ *which is either*
   - $(r(d+r)^{\omega_2/2}n\log q + r^{\lambda}n(d+m)/m\log q + n\log^2 q)^{1+o(1)}$ *if* $d < n$
   - *or* $(r(d+r)n^{\omega-1}\log q + r^{\lambda}n(d+m)/m\log q + n\log^2 q)^{1+o(1)}$ *otherwise*
3. $((r^{\lambda}/m + r^{\omega}/\sqrt{m})n^2\log q + n\log^2 q)^{1+o(1)}$ *if* $u = \tau^n$
4. $(r^{\omega}n^{1.5}\log q + n\log^2 q)^{1+o(1)}$ *if* $u = \tau^n$ *and* $\mathbb{L} = \mathbb{F}_{\mathfrak{p}}$.

For Drinfeld $\mathbb{F}_q[x]$-modules, recall $W(\mathbb{L}) \cong \mathbb{L}[[z - \gamma_x]]$. Moreover, $H^*_{\mathrm{crys}}(\phi, \mathbb{L}) \cong \mathbb{L}\{\tau\}\tau$ under the identification $\eta \mapsto \eta_x$, with the $W(\mathbb{L})$-action given by $x * s = s\phi_x$ for $s \in \mathbb{L}\{\tau\}\tau$. Therefore, for the remainder of this section we will state our algorithms concretely with elements from $\mathbb{L}\{\tau\}\tau$. In this case, we have a standard $W(\mathbb{L})$-basis consisting of skew polynomials $\mathfrak{b} = \{\tau, \ldots, \tau^r\}$, and a skew polynomial u corresponding to an endomorphism of ϕ of degree d. For any skew polynomial $s \in \mathbb{L}\{\tau\}\tau$, if $s = \mu_1\tau^1 + \ldots + \mu_r\tau^r$ with

$\mu_i \in W(\mathbb{L})$, then let $\mu(s)$ denote its coefficient vector with respect to the standard basis, that is

$$\mu(s) = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_r \end{bmatrix},$$

and let $\mu^{(i)} = \mu(\tau^i)$.

Our procedure for computing CharPoly(u) will resort to computing a matrix representation $\mathfrak{m}(u)$ of the action of $\hat{u}$ on $H^*_{\text{crys}}(\phi, \mathbb{L})$ with respect to $\mathfrak{b}$ with coefficients in W_k for $k > \frac{d}{m}$. Concretely, we compute the matrix $\mathfrak{m}(u) = \begin{bmatrix} \mu(\tau u) & \dots & \mu(\tau^r u) \end{bmatrix}$ with $(i,j)^{th}$ entries $\mu_{i,j}$ truncated to precision k. We may then apply $\chi_k : W_k \to \mathbb{F}_q[x]/\mathfrak{p}^k$ to the coefficients of CharPoly$(\mathfrak{m}(u))$ to recover the coefficients of CharPoly(u).

6.4.1 A Divide-and-conquer Approach

In this section, we will construct and analyze algorithm 2 of theorem 6.4.1. Namely, that there is a deterministic algorithm to compute the characteristic polynomial of a degree d endomorphism u with bit complexity $(r\text{SM}(d+r, n, q) + r^\lambda n(d+m)/m \log q + n \log^2 q)^{1+o(1)}$. If we are given a factorization of the form

$$\tau^j u = \sum_{\ell=0}^{K} (f_{\ell,1}\tau^1 + \ldots + f_{\ell,r}\tau^r)\phi_x^j$$

with $f_{\ell,i} \in \mathbb{L}$ then we may recover the coefficients $\mu_{i,j}$ using

$$\mu_{i,j} = \sum_{\ell=0}^{K} f_{\ell,j} x^\ell \tag{6.15}$$

We have that $K \leq \lfloor \frac{d}{r} \rfloor$. By performing skew-polynomial Euclidean division of $\tau^j u$ by $\phi_x^{\lfloor K/2 \rfloor}$, we obtain a decomposition

$$\tau^j u = g\phi_x^{\lfloor K/2 \rfloor} + h$$

To which we may recursively apply our decomposition algorithm until $\deg g, \deg h \leq r$. Using fast exponentiation, we can compute ϕ_x^i for $O(\log d)$ values i for a cost of $O(\mathrm{SM}(d, n, q))$, and perform Euclidean division of degree at most $d+r$ for $O(\mathrm{SM}(d+r, n, q))$ cost.

Using the Puchinger and Wachter-Zeh, the complexity of the algorithm becomes $(r(d+r)^{\omega_2/2} n \log q + r^{\lambda} n(d+m)/m \log q + n \log^2 q)^{1+o(1)}$, and is the preferred approach when $d \in O(n)$. Instead, using the Caruso and Le Borgne algorithm, we obtain a complexity of $(r(d+r)n^{\omega-1} \log q + r^{\lambda} n(d+m)/m \log q + n \log^2 q)^{1+o(1)}$, which is better for large d.

6.4.2 Methods using a Recurrence

The following lemma will be useful in allowing us to compute the action of endomorphisms on $\mathbb{L}\{\tau\}\tau$:

Lemma 6.4.2. *Let ϕ be a rank r Drinfeld $\mathbb{F}_q[x]$-module defined by $\phi_x = \sum_{i=0}^{r} \Delta_i \tau^i$. For any $t \geq 1$, the following relation holds in the $W(\mathbb{L})$-module $\mathbb{L}\{\tau\}\tau$:*

$$\sum_{i=0}^{r} \Delta_i^{[t]} \tau^{t+i} = z * \tau^t. \tag{6.16}$$

Proof. The result follows from applying the definition module action of $\mathbb{L}[z]$ on $\mathbb{L}\{\tau\}\tau$ to $z * \tau^t$.

$$\begin{aligned} z * \tau^t = \tau^t \phi_x &= \tau^t \sum_{i=0}^{r} \Delta_i \tau^i \\ &= \sum_{i=0}^{r} \Delta_i^{[t]} \tau^{t+i}. \end{aligned}$$

□

For $0 \leq i < r$, set $\Lambda_i = -\frac{\Delta_i}{\Delta_r}$. For any $t > 0$, we define a matrix $\mathcal{A} \in M_r(\mathbb{L}[z])$ to be:

$$\mathcal{A} = \begin{bmatrix} \Lambda_{r-1} & \Lambda_{r-2} & \dots & \Lambda_1 & \Lambda_0 + \frac{z}{\Delta_r} \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix} \tag{6.17}$$

and set $\mathcal{A}_t \in M_r(\mathbb{L}[z])$ to be

$$\mathcal{A}_t = \mathcal{A}^{[t]} = \begin{bmatrix} \Lambda_{r-1}^{[t]} & \Lambda_{r-2}^{[t]} & \dots & \Lambda_1^{[t]} & \Lambda_0^{[t]} + \frac{z}{\Delta_r^{[t]}} \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}. \tag{6.18}$$

Recalling that the operator $\cdot^{[t]}$ indicates Frobenius σ_q^t applied coefficient-wise for polynomials, and entry-wise for matrices. For any skew polynomial $s \in \mathbb{L}\{\tau\}\tau$, let $\mu(s)$ denote its coefficient vector with respect to the standard basis and let $\mu(s)_j$ denote the coefficient of $\tau^j \in \mathfrak{b}$. If we let $\mu^{(i)} = \mu(\tau^i)$, then:

$$\begin{bmatrix} \mu^{(t+r)} \\ \mu^{(t+r-1)} \\ \vdots \\ \mu^{(t+1)} \end{bmatrix} = \mathcal{A}_t \begin{bmatrix} \mu^{(t+r-1)} \\ \mu^{(t+r-2)} \\ \vdots \\ \mu^{(t)} \end{bmatrix}. \tag{6.19}$$

By reducing the entries modulo $g_k = (z - \gamma_x)^k$, we obtain the coefficients for the basis expansion in W_k. For general endomorphisms u, we can compute $\mathfrak{m}(u)$ with coefficients truncated to precision k as follows: write $u = \sum_{t=0}^{d} u_t \tau^t \in \operatorname{End}(H^*_{\text{crys}}(\phi, \mathbb{L}))$ of degree d over W_k. For each $1 \le i \le r$, we have

$$\tau^i u = u_0^{[i]} \tau^i + \dots + u_d^{[i]} \tau^{d+i}.$$

Our goal is to compute the matrix:

$$\mathfrak{m}(u) = \begin{bmatrix} \mu(\tau u) & \mu(\tau^2 u) & \dots & \mu(\tau^r u) \end{bmatrix}$$

It suffices to first compute basis expansions to obtain $\mu^{(j)}$ for $j \leq d + r$ using equation (6.19), and then compute vectors $\mu(\tau^i u)$ using $\mu(\tau^i u)_j = \sum_{t=0}^{d} u_t \mu_j^{(i+t)}$.

Preparing the coefficients $\mathcal{A}_1, \dots, \mathcal{A}_d$ costs $O(dr)$ field operations and applications of the Frobenius, contributing a bit cost of $(drn \log q)^{1+o(1)}$. Each application of the recurrence from (6.16) incurs a bit cost of $(kr^2 n \log q)^{1+o(1)}$ from scaling operations by coefficients of the form $\Lambda_i^{[t]}$ as well as a cost of $(dkrn \log q)^{1+o(1)}$ from scaling and modular reduction from the linear term $\Lambda_0^{[t]} + \frac{z}{\Delta_r^{[t]}}$.

Recovering the coefficients of $\mu(\tau^i u)$ modulo g_k takes $O(dkr)$ operations in $\mathbb{L}$. In the case where we are interested in computing the characteristic polynomial, it is sufficient that $k > \frac{d}{m}$ to allow for the recovery of the coefficients inside of $\mathbb{F}_q[x]/\mathfrak{p}^k$ after applying χ_k. In this case, the complexity for using the recurrence to compute $\mu^{(j)}$ becomes

$$((d + r)(d + m)rn/m \log q)^{1+o(1)}$$

and for recovering the coefficients of the matrix, we have a cost of

$$(d(d + m)r^2 n/m \log q)^{1+o(1)}.$$

At this stage, we have our matrix $\mathfrak{m}(u) \in M_r(W_k)$; computing the characteristic polynomial of $\mathfrak{m}(u)$ costs $r^\lambda kn \log q$. We also incur a cost of $n \log^2 q$ to pre-compute polynomial representations of all Frobenius operators $\{\sigma_q, \dots, \sigma_q^{n-1}\}$ for use in modular composition. Summing these costs gives us the complexity of algorithm 1 of theorem 6.4.1.

6.4.3 A Baby-step Giant-step Algorithm to Compute the Characteristic Polynomial of the Frobenius Endomorphism

We will now describe an algorithm for computing the characteristic polynomial of the Frobenius endomorphism τ^n based on the recurrence of lemma (6.16), which is the basis of algorithms 3 and 4 of theorem 6.4.1. Building on the approach of the previous section, when the endomorphism in question is the Frobenius, we are only interested in computing the vectors $\mu^{(n+1)}, \ldots, \mu^{(n+r)}$, which we can write as

$$\begin{bmatrix} \mu^{(n+r)} \\ \mu^{(n+r-1)} \\ \vdots \\ \mu^{(n+1)} \end{bmatrix} = \mathcal{A}_n \ldots \mathcal{A}_1 = \mathcal{A}, \tag{6.20}$$

Since we only need the final product, we can compute it using a baby-step giant-step approach inspired by a technique in [21]. Write $n^* = \lceil \sqrt{nk} \rceil \in O(n/\sqrt{m})$, and let n be written as $n = n^* n_1 + n_0$ with $0 \le n_0 < n^*$, so that $n_1 \le \sqrt{n/k}$. Now let:

$$\mathcal{C}_0 = \mathcal{A}^{[n_0]} \ldots \mathcal{A}^{[1]}$$

and

$$\mathcal{C} = \mathcal{A}^{[n^*+n_0]} \cdots \mathcal{A}^{[n_0+1]}.$$

Then we have:

$$\mathcal{A}_* = \mathcal{C}^{[(n_1-1)n^*]} \cdots \mathcal{C}^{[n^*]} \mathcal{C} \mathcal{C}_0.$$

To obtain the desired complexity, we will seek to work with matrices with coefficients reduced modulo g_k, and in particular we want to compute

$$\mathcal{A}_* \bmod g_k = (\mathcal{C}^{[(n_1-1)n^*]} \bmod g_k) \cdots (\mathcal{C}^{[n^*]} \bmod g_k)(\mathcal{C} \bmod g_k)(\mathcal{C}_0 \bmod g_k).$$

To do this, we will need the following lemma

Lemma 6.4.3. *For f, g in $\mathbb{L}[x]$ and $t \ge 0$,*

$$f^{[t]} \bmod g = (f \bmod g^{[-t]})^{[t]}$$

Proof. Let $R \in \mathbb{L}[x]$ such that $\deg R < \deg g$ and $f = Qg^{[-t]} + R$ for some $Q \in \mathbb{L}[x]$. Then $f^{[t]} = Q^{[t]}g + R^{[t]}$ while $\deg R^{[t]} < \deg g$, from which we can conclude the lemma. $\square$

Using lemma 6.4.3, we can compute $\mathcal{C}^{[in^*]} \bmod g_k = (\mathcal{C} \bmod g_k^{[-in^*]})^{[in^*]}$ for $0 \leq i < n_1$. Computing matrices $\mathcal{A}^{[1]}, \ldots, \mathcal{A}^{[n^*+n_0]}$ costs $O(nr/\sqrt{m})$ field operations and applications of the Frobenius, contributing a total bit cost of $(rn^2rm^{-1/2}\log q)^{1+o(1)}$. Computing $\mathcal{C}$ and $\mathcal{C}_0$ can be done with cost $(r^\omega n^2/\sqrt{m}\log q)^{1+o(1)}$ using classical techniques [30]. Each entry in $\mathcal{C}$ has degrees of order $O(n/\sqrt{m})$; computing $\mathcal{C}^{[in^*]} \bmod g_k$ takes $O(r^2n/\sqrt{m})$ applications of the Frobenius and $O(r^2)$ polynomial euclidean divisions of degree at most $O(n/\sqrt{m})$. These tasks together cost a total of $(r^2n^2/\sqrt{m}\log q)^{1+o(1)}$ bit operations. The final step towards computing $\mathcal{A}_*$ involved taking the matrix product $(\mathcal{C}^{[(n_1-1)n^*]} \bmod g_k)\cdots(\mathcal{C}^{[n^*]} \bmod g_k)(\mathcal{C} \bmod g_k)(\mathcal{C}_0 \bmod g_k)$. This takes $(r^\omega n^2/\sqrt{m}\log q)^{1+o(1)}$ bit operations.

Having computed $\mathcal{A}_*$, the remainder of the algorithm consists of computing its characteristic polynomial, and applying the map χ_k to its coefficients. When $k = 1$, $W_k \cong \mathbb{L}$ and in the computation of the characteristic polynomial can be done in $O(r^\omega)$ $\mathbb{L}$-operations. For other values of k, computing the characteristic polynomial takes $O(r^\lambda)$ operations in W_k, with λ the coefficient defined in section 2.2.2. This contributes a bit cost of $(r^\lambda kn\log q)^{1+o(1)}$. Summing the costs gives the complexities of algorithm 3 and 4 of theorem 6.4.1.

Algorithm 3 Characteristic Polynomial of the Frobenius

```
**Input** A Drinfeld $\mathbb{F}_q[x]$-module $\phi_x = \sum_{i=0}^{r} \Delta_i \tau^i$
**Output** A degree $r$ polynomial in $\mathbb{F}_q[x][Z]$

1: $n^*, n_1, n_0 \leftarrow \lceil \sqrt{nk} \rceil, \lfloor n/n^* \rfloor, n \bmod n^*$.
2: $\mathcal{A}$ as in (6.17)
3: $\mathcal{C} \leftarrow \mathcal{A}^{[n^*+n_0]} \dots \mathcal{A}^{[n_0+1]}$.
4: $\mathcal{C}_0 \leftarrow \mathcal{A}^{[n_0]} \dots \mathcal{A}^{[1]} \bmod g$
5: $\mathcal{C}^{[in^*]} \leftarrow (\mathcal{C} \bmod g^{[-in^*]})^{[in^*]}$ for $0 \leq i < n_1$.
6: $\mathcal{A}_* \leftarrow \left( \prod_{i=0}^{n_1-1} \bar{\mathcal{C}}^{[in^*]} \right) \bar{\mathcal{C}}_0$
7: $\bar{a}_i \leftarrow$ coefficient of $Z^i$ in $\det(\mathcal{A}_* - ZI)$
8: **return** $a_i = \chi_k(\bar{a}_i)$ for $0 \leq i < r$
```

6.4.4 Timings

The algorithm of section 6.4.3 has been implemented and merged into the SageMath [72] open source software system as part of the implementation of Drinfeld modules. A MAGMA [7] implementation of the same algorithm is available at https://github.com/ymusleh/drinfeld-magma and was used to generate the timings below

	$n = 100$	$n = 150$	$n = 200$	$n = 300$	$n = 400$	$n = 500$	$n = 600$
$r = 5$	0.400	2.260	42.190	86.830	269.760	635.170	1099.110
$r = 9$	0.790	4.210	78.860	157.100	481.090	1129.670	
$r = 12$	1.170	6.080	104.630	220.430	658.950	1531.580	
$r = 18$	2.300	11.360	170.790	366.690	1074.840	2451.530	
$r = 23$	3.820	17.580	240.100	525.670	1518.370		

Table 6.1: Run Times for $m = 10$ $q = 25$ in seconds

We will note that the results here differ from the theoretical predictions of the performance of the algorithm. The main reason for this discrepancy is the apparent lack of an implementation of Kedlaya-Umans modular composition, which impacts the cost of computing Frobenius powers. We will also note that for computing images under the map χ_k, we used a simpler technique based on Gröbner bases rather than the tangling algorithm described here.

Chapter 7

Additional Algorithms

We will give an analysis of several algorithms connected to skew polynomials and Drinfeld modules. In particular, we will analyze a new approach for both multi-point evaluation and minimal subspace polynomial combining the mutually recursive approach of Puchinger and Wachter-Zeh with a Horner-type scheme. This algorithm is used in the complexity analysis of theorem 6.2.1.

Finally, we will describe the algorithm of Wesolowski given in [74] for computing an explicit basis over $\mathbb{F}_q$ of the space of morphisms $\mathrm{Hom}(\phi, \psi)_d$ of degree at most d between arbitrary pairs of Drinfeld modules ϕ, ψ. Moreover, we will describe an extension to this approach, which yields algorithms to compute a basis for $\mathrm{Hom}(\phi, \psi)$ as a free module over both $\mathbb{F}_q[\tau^n]$ and $\mathbb{F}_q[x]$, as well as provide a complexity analysis for all algorithms discussed.

7.1 An Algorithm for Multi-point Evaluation of Skew Polynomials

Let $\mathrm{End}^V_{\mathbb{F}}(U)$ denote the set of $\mathbb{F}$-vector space endomorphisms on an $\mathbb{F}$-vector space U. We consider a tower of finite fields $\mathbb{F}_q \subset \mathbb{L}$ such that $[\mathbb{L} : \mathbb{F}_q] = n$. If σ is an $\mathbb{F}_q$-linear operator on the n-dimensional vector space $\mathbb{L}$, then we have a mapping, the *evaluation* map $\mathrm{eval} : \mathbb{L}\{\tau; \sigma\} \to \mathrm{End}^V_{\mathbb{F}_q}(\mathbb{L})$ determined by, for all $a \in \mathbb{L}$:

$$\operatorname{eval}\Big(\sum_i s_i\tau^i\Big)(a) = \sum_i s_i\sigma^i(a) \tag{7.1}$$

To simplify notation, we will let $s(a) = \operatorname{eval}(s)(a)$. Recall that given t evaluation points $a_1, \ldots, a_t$ and a degree d skew polynomial $s = \sum_{i=0}^{d} s_i\tau^i \in$ the multi-point evaluation problem asks to compute $s(a_1), \ldots, s(a_t)$. We now present an algorithm to compute the multi-point evaluation of s at $a_1, \ldots, a_t$ which originally appeared in [56].

Theorem 7.1.1. *Given a skew polynomial $s \in \mathbb{L}\{\tau\}$ of degree d and t evaluation points $a_1, \ldots, a_t \in \mathbb{L}$, there exists an algorithm for computing the evaluations $s(a_1), \ldots, s(a_t)$ in time*

- $(dt^{\omega-2}n\log q)^{1+o(1)}$ *if $t \leq \sqrt{d}$.*
- $(d^{(\omega-1)/2}tn\log q)^{1+o(1)}$ *otherwise.*

Proof. Without loss of generality assume $d+1$ is a perfect square and set $\delta = \sqrt{d+1}$. For $S_1, \ldots, S_{\delta-1} \in \mathbb{L}\{\tau\}$, write:

$$s = S_0 + \tau^{\delta}S_1 + \ldots + \tau^{\delta(\delta-1)}S_{\delta-1}$$

where $\deg S_i < \delta$ and $S_i = \sum_{j=0}^{\delta} s_{i,j}\tau^j$. Next, compute $a_{i,j} = \sigma^i(a_j)$. Furthermore, we have that:

$$\begin{bmatrix} s_{0,0} & \cdots & s_{0,\delta-1} \\ \vdots & \ddots & \vdots \\ s_{\delta-1,0} & \cdots & s_{\delta-1,\delta-1} \end{bmatrix} \begin{bmatrix} a_{0,1} & a_{0,2} & \cdots & a_{0,t} \\ \vdots & \vdots & \ddots & \vdots \\ a_{\delta-1,1} & a_{\delta-1,2} & \cdots & \alpha_{s-1,t} \end{bmatrix} = \begin{bmatrix} S_0(a_1) & \cdots & S_0(a_t) \\ \vdots & \ddots & \vdots \\ S_{\delta-1}(a_1) & \cdots & S_{\delta-1}(a_t) \end{bmatrix} \tag{7.2}$$

Finally, we can compute the evaluations $S(a_j)$ using a Horner scheme $S(a_j) = S_0(a_j) + \tau^{\delta}(S_1(a_j) + \tau^{\delta}(S_2(a_j) + \ldots))$.

The dominant cost of this procedure is computing the $\delta \times \delta$ and $\delta \times t$ matrix product of equation (7.2) which costs $O(dt^{\omega-2})$ $\mathbb{L}$-operations if $t \leq \sqrt{d+1}$, and $d^{(\omega-1)/2}t$ otherwise. The complexity claims follow. $\square$

We observe that in the "small t" case with $t \leq \sqrt{d}$, the complexity of the algorithm is $(d^{\omega/2} n \log q)^{1+o(1)}$. This compares favourably with the Puchinger and Wachter-Zeh approach discussed in section §2.4.3, which has a complexity of $(d^{\max(\log_2(3),\omega_2/2)} n \log q)^{1+o(1)}$ in the same setting.

Algorithm 4 Multipoint Evaluation [56]

Input A skew polynomial $s \in \mathbb{L}\{\tau\}$ of degree d, and a set of linearly independent evaluation points $\{a_1, \dots, a_t\} \subset \mathbb{L}$.

Output The set $\{s(a_1), \dots, s(a_t)\} \subset \mathbb{L}$

```
1: Compute $s_i = s_{i,0} + s_{i,1}\tau + \dots + s_{i,\delta-1}\tau^{\delta}$ such that $s = s_0 + \tau^{\delta} s_1 + \dots + \tau^{\delta(\delta-1)} s_{\delta-1}$
2: Compute $a_{i,j} = \sigma^i(a_j)$ for each $0 \leq i \leq \delta - 1$ and $1 \leq j \leq t$
3: Compute the $\delta \times \delta$ and $\delta \times t$ matrix product:
   $\begin{bmatrix} s_{0,0} & \dots & s_{0,\delta-1} \\ \vdots & & \vdots \\ s_{\delta-1,0} & \dots & s_{\delta-1,\delta-1} \end{bmatrix} \begin{bmatrix} a_{0,0} & \dots & a_{0,t} \\ \vdots & & \vdots \\ a_{\delta-1,0} & \dots & a_{\delta-1,t} \end{bmatrix} = \begin{bmatrix} s_0(a_1) & \dots & s_0(a_t) \\ \vdots & & \vdots \\ s_{\delta-1}(a_1) & \dots & s_{\delta-1}(a_t) \end{bmatrix}$
4: For $1 \leq j \leq t$ compute $s(a_j) = s_0(a_j) + \tau^{\delta}(s_1(a_j) + \tau^{\delta}(s_2(a_j) + \dots$ via Horner's scheme
```

7.2 Minimal Subspace Polynomial

Given $t \leq n$ elements $a_1, \dots, a_t \in \mathbb{L}$ linearly independent over $\mathbb{F}_q$, recall that the minimal subspace polynomial problem asks to compute the skew polynomial s of minimal degree such that $s(a_i) = 0$ for each $i \leq t$. The mutually recursive algorithms of [64] allow us to convert any algorithm for either multi-point evaluation of minimal subspace polynomial into an algorithm for the other. In particular, we recall that the recurrence for minimal subspace has the form

$$\mathrm{MSP}(d) = 2\mathrm{MSP}(d/2) + \mathrm{MPE}(d/2, d/2) + \mathrm{SM}(d/2, n, q).$$

Corollary 7.2.1. *Let* $v_1, \dots, v_d \in \mathbb{L}$. *There is an algorithm to compute* $\mathrm{MSP}(\{v_1, \dots, v_d\}) \in \mathbb{L}\{\tau\}$ *with a bit complexity of* $(d^{(\omega+1)/2} n \log q)^{1+o(1)}$.

Proof. The complexity result of theorem 7.1.1 give a recurrence of the form

$$\mathrm{MSP}(d) = 2\mathrm{MSP}(d/2) + (d^{(\omega+1)/2} n \log q)^{1+o(1)} + \mathrm{SM}(d/2, n, q),$$

yielding an overall bit complexity of $\mathrm{MSP}(d) = (d^{(\omega+1)/2} n \log q)^{1+o(1)}$.

□

7.3 Computing Endomorphism Rings of Drinfeld Modules

In the classical setting, the structure of the endomorphism ring of elliptic curves plays an important role in several areas, including class field theory via the correspondence between CM elliptic curves and orders in imaginary quadratic number fields. Isogenies also play an important role in point counting algorithms and in classical cryptography, including possible post-quantum schemes, over elliptic curves, making their explicit computation is an important mechanism. Important results in this direction include Vélu's formula for computing, given a base elliptic curve E and a subgroup G, computes an elliptic curve E' and isogeny $t : E \to E'$ whose kernel is exactly G [73]. A related problem is the *explicit isogeny problem* which, given j-invariants j, j', determines if elliptic curves belonging to the corresponding isomorphism class are isogenous of degree d, and if so, computes curves E, E' and an isogeny $t : R \to E'$. This is a very well studied problem for elliptic curves, and several algorithms have been proposed including in [27, 17, 53, 18].

The structure of the set of morphisms between Drinfeld modules, including the special case of the endomorphism ring and the explicit computation of bases, has been studied in several recent papers, including [51, 29]. In [74], an algorithm for explicitly computing an isogeny between fixed Drinfeld modules was given. Here, we will give a complexity analysis of several algorithms that largely follow the methodology of the approaches discussed in [51, 74]; the main contribution here is the complexity analysis, and in the case of the basis for $\mathrm{Hom}(\phi, \psi)$ over $\mathbb{F}_q[x]$, the use of the recurrence method to compute explicit basis expansions.

Recall that $\mathrm{Hom}(\phi, \psi) = \{u \in \mathbb{L}\{\tau\} \mid u\phi_x = \psi_x u\}$. In particular, for all ϕ, ψ:

- $\mathrm{Hom}(\phi, \psi)_d$ is a vector space over $\mathbb{F}_q$ of dimension at most dn.
- $\mathrm{Hom}(\phi, \psi)$ is a free module over $\mathbb{F}_q[\tau^n]$ of dimension at most n^2.
- $\mathrm{Hom}(\phi, \psi)$ is a free module over $\mathbb{F}_q[x]$, where $x * u = u\phi_x = \psi_x u$, of dimension at most r^2 [33], and as we shall see, also of dimension at most nr.

A more natural way to view the problem is to consider the set $\mathbb{L}\{\tau\}$ as a module over the corresponding coefficient rings $\mathbb{F}_q$, $\mathbb{F}_q[\tau^n]$, and $\mathbb{F}_q[x]$. In each case, $\mathrm{Hom}(\phi, \psi)$ is a linear subspace of $\mathbb{L}\{\tau\}$ given by the kernel of the linear operator $u \mapsto u\phi_x - \psi_x u$. To compute a basis for $\mathrm{Hom}(\phi, \psi)$, it therefore suffices to compute an explicit matrix for the action of this operator with respect to a canonical basis for $\mathbb{L}\{\tau\}$ over each coefficient ring, and compute its kernel. The results of this section are summarized in the following theorem.

Theorem 7.3.1. *There is an algorithm for computing a basis of*

1. $\mathrm{Hom}(\phi, \psi)_d$ *over* $\mathbb{F}_q$ *using* $(d^\omega n^\omega \log q + dn^2 r \log q + n \log^2 q)^{1+o(1)}$ *bit operations.*
2. $\mathrm{Hom}(\phi, \psi)$ *over* $\mathbb{F}_q[\tau^n]$ *using* $(n^{2\omega} r \log q + n \log^2 q)^{1+o(1)}$ *bit operations.*
3. $\mathrm{Hom}(\phi, \psi)$ *over* $\mathbb{F}_q[x]$ *using* $(n^3 r^3 \log q + n \log^2 q)^{1+o(1)}$ *bit operations.*

Proof. A polynomial time algorithm for computing an $\mathbb{F}_q$-basis for $\mathrm{Hom}(\phi, \psi)_d$ using the approach described above was first given in [74]. We will describe this algorithm below, and analyze its complexity.

Fix a basis $\{f_j\}_{j=0}^{n-1}$ for $\mathbb{L}/\mathbb{F}_q$. Set $\phi_x = \sum_{i=0}^r \Delta_i \tau^i$, $\psi_x = \sum_{i=0}^r \Lambda_i \tau^i$, and $u = \displaystyle\sum_{j=1}^{n}\sum_{k=0}^{d} u_{j,k} f_j \tau^i$, with $u_{j,k} \in \mathbb{F}_q$. Then $u \in \mathrm{Hom}(\phi, \psi)_d = \{u \in \mathrm{Hom}(\phi, \psi) \mid \deg_\tau(u) \le d\}$ if and only if it satisfies the following relation:

$$\sum_{i=0}^{r}\sum_{j=0}^{n-1}\sum_{k=0}^{d} \Delta_i^{[k]} f_j u_{j,k} \tau^{i+k} = \sum_{i=0}^{r}\sum_{j=1}^{n}\sum_{k=0}^{d} \Lambda_i u_{j,k} f_j^{[i]} \tau^{i+k}. \tag{7.3}$$

We can compute all products $\Lambda_i f_j^{[i]}$, $\Delta_i^{[k]} f_j$ for a total bit cost of $(dn^2 r \log q)^{1+o(1)}$. Rewriting each entry in terms of the $\mathbb{F}_q$-basis for $\mathbb{L}$

$$\Lambda_i f_j^{[i]} = \sum_{\theta=0}^{n-1} \lambda_{i,j,\theta} f_\theta$$
$$\Delta_i^{[k]} f_j = \sum_{\theta=0}^{n-1} \delta_{i,j,k,\theta} f_\theta$$

allows us to define a linear system over $\mathbb{F}_q$ coming from

$$\sum_{i=0}^{r}\sum_{j=0}^{n-1}\sum_{k=0}^{d}\sum_{\theta=0}^{n-1} \delta_{i,j,k,\theta} u_{j,k} f_\theta \tau^{i+k} = \sum_{i=0}^{r}\sum_{j=1}^{n}\sum_{k=0}^{d}\sum_{\theta=0}^{n-1} \lambda_{i,j,\theta} u_{j,k} f_\theta \tau^{i+k}. \tag{7.4}$$

We therefore obtains a linear equation in the $\mathbb{F}_q$ variables $u_{j,k}$ from extracting the coefficients of each $f_j \tau^i$ for $0 \leq j < n$ and $0 \leq i \leq d+r$, allowing us to construct a $(d+r)n \times (d+1)n$ linear system over $\mathbb{F}_q$ whose kernel is exactly $\mathrm{Hom}(\phi, \psi)_d$. Constructing the system from equation (7.4) costs $O(dn^2 r)$ operations in $\mathbb{F}_q$ as well as $O(r+n)$ applications of the Frobenius. Pre-computing the Frobenius takes the usual $(n \log^2 q)^{1+o(1)}$ bit cost; with the overall bit cost to construct and solve the system being $(d^\omega n^\omega \log q + dn^2 r \log q)^{1+o(1)}$, we obtain the complexity stated in item 1 of theorem 7.3.1.

One can repeat a similar procedure for bases of $\mathrm{Hom}(\phi, \psi)$ over $\mathbb{F}_q[\tau^n]$ and $\mathbb{F}_q[x]$, and indeed such a computation was discussed in [51]. In the former case, $\{f_j \tau^k\}_{1 \leq j \leq n}^{0 \leq k < n}$ is an $\mathbb{F}_q$-basis for $\mathbb{L}\{\tau\}$ with coefficients in $\mathbb{F}_q[\tau^n]$; so we can write $u = \sum_{j=1}^{n}\sum_{k=0}^{n-1} \alpha_{j,k} f_j \tau^k$ with $\alpha_{i,j} \in \mathbb{F}_q[\tau^n]$. Then $\mathrm{Hom}(\phi, \psi)$ consists of such u satisfying the relation

$$\sum_{i=0}^{r}\sum_{j=0}^{n-1}\sum_{k=0}^{n-1} \Delta_i^{[k]} f_j \alpha_{j,k} \tau^{i+k} = \sum_{i=0}^{r}\sum_{j=0}^{n-1}\sum_{k=0}^{n-1} \Lambda_i \alpha_{j,k} f_j^{[i]} \tau^{i+k} \tag{7.5}$$

As seen previously, we can compute a basis decomposition of all $\Lambda_i f_j^{[i]}$ and $\Delta_i^{[k]} f_j$ at a bit cost of $(rn^3 \log q)^{1+o(1)}$. Then equation 7.6 becomes:

$$\sum_{i=0}^{r}\sum_{j=0}^{n-1}\sum_{k=0}^{n-1}\sum_{\theta=0}^{n-1}\delta_{i,j,k,\theta}\alpha_{j,k}\tau^{i+k} = \sum_{i=0}^{r}\sum_{j=0}^{n-1}\sum_{k=0}^{n-1}\sum_{\theta=0}^{n-1}\lambda_{i,j,\theta}\alpha_{j,k}\tau^{i+k} \tag{7.6}$$

By rewriting the above expression in terms of the basis elements $f_j\tau^k$ for $j < n$ and $k < n$ with coefficients $\rho_{i,j,k,\ell} \in \mathbb{F}_q[\tau^n]$, we obtain:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\sum_{k=0}^{n-1}\sum_{\ell=0}^{n-1}\rho_{i,j,k,\ell}\alpha_{k,\ell}f_j\tau^i = 0$$

Then the coefficients $\alpha_{k,\ell}$ of u can be solved for by constructing an $n^2 \times n^2$ linear system over $\mathbb{F}_q[\tau^n]$, whose coefficients have degree at most $\lfloor \frac{r}{n} \rfloor + 1$, of the form

$$\begin{bmatrix} \rho_{0,0,0,0} & \cdots & \rho_{0,0,n-1,n-1} \\ \vdots & \ddots & \vdots \\ \rho_{n-1,n-1,0,0} & \cdots & \rho_{n-1,n-1,n-1,n-1} \end{bmatrix} \begin{bmatrix} \alpha_{0,0} \\ \vdots \\ \alpha_{n-1,n-1} \end{bmatrix} = 0$$

We may then take advantage of pre-existing algorithms for computing a minimal nullspace basis for the system with a complexity of $O(n^{2\omega-1}r)$ field operations in $\mathbb{F}_q$ [76], adding a cost of $(n^{2\omega}r\log q)$ in our bit complexity model. Constructing the system costs at most $(n^3r\log q)^{1+o(1)}$ bit operations, and adding in the usual pre-computation for Frobenius maps gives the stated complexity in item 2 of theorem 7.3.1.

We can repeat the core idea of this algorithm once again for a basis over $\mathbb{F}_q[x]$, though some more care must be taken to address technical details. More explicitly, we view $\mathbb{L}\{\tau\}$ as an $\mathbb{F}_q[x]$-module with the action $x * u = u\phi_x$, and note that u is not required to lie inside $\mathrm{Hom}(\phi, \psi)$. Write $u = \sum_{i=0}^{r-1}\sum_{j=0}^{n-1} u_{i,j}f_j\tau^i$ with $u_{i,j} \in \mathbb{F}_q[x]$. Then if $u \in \mathrm{Hom}(\phi,\psi)$ we must have

$$u\phi_x = x * u = \sum_{i=0}^{r-1}\sum_{j=0}^{n-1} xu_{i,j}f_j\tau^i = \psi_x u = \sum_{i=0}^{r-1}\sum_{j=0}^{n-1}\sum_{k=0}^{r}\Lambda_k f_j^{[k]}u_{i,j}\tau^{i+k} \tag{7.7}$$

Constructing the linear system as we did previously can be done by computing coefficients $\rho_{t,i,j} \in \mathbb{F}_q[x]$ such that

$$\tau^t = \sum_{w=0}^{r-1} \sum_{z=0}^{n-1} \rho_{t,w,z} f_z \tau^w$$

for $r \leq t < 2r$. Recalling the definition of the $\mathbb{F}_q[x]$-action on $\mathbb{L}\{\tau\}$ $x\tau^i = \tau^i \phi_x$, we can re-arrange the expression to obtain the following recurrence, for any $i \geq 0$

$$\tau^{r+i} = \sum_{k=0}^{r-1} \left(-\frac{\Delta_k}{\Delta_r} \right)^{[i]} \tau^{k+i} + x\tau^i. \tag{7.8}$$

This can be evaluated at most $r-1$ times to allow the extraction of the required coefficients $\rho_{t,i,j}$. Note that $\deg \rho_{t,i,j} \leq 1$. Each such evaluation requires $O(r)$ applications of the Frobenius, and $O(r^2)$ operations in $\mathbb{L}$. This gives an overall bit complexity of $(r^3 n \log q)^{1+o(1)}$ for computing all required $\rho_{t,i,j}$. Then equation (7.7) yields:

$$\sum_{i=0}^{r-1} \sum_{j=0}^{n-1} x u_{i,j} f_j \tau^i = \sum_{i=0}^{r-1} \sum_{j=0}^{n-1} \sum_{k=0}^{r} \sum_{w=0}^{r-1} \sum_{z=0}^{n-1} \Lambda_k f_j^{[k]} u_{i,j} \rho_{i+k,w,z} f_z \tau^w \tag{7.9}$$

In this case, it suffices to compute the decomposition of $\Lambda_k f_j^{[k]} = \sum_{\theta=0}^{n-1} \lambda_{k,j,\theta} f_\theta$ for $k \leq r$, $j \leq n$ for a total bit cost of $(n^2 r \log q)^{1+o(1)}$. Using the above equation, we can determine the coefficients of the basis elements $f_z \tau^w$ in terms of the indeterminates u_{ij}, which allows the extraction of an $nr \times nr$ linear system, with entries in $\mathbb{F}_q[x]_1$, in the usual manner. This system can then be solved over $\mathbb{F}_q[x]$ with a bit complexity of $(n^\omega r^\omega \log q)^{1+o(1)}$. Generating the system from (7.9) costs $(r^3 n^3 \log q)^{1+o(1)}$, which gives an overall cost of $(r^3 n^3 \log q + n \log^2 q)^{1+o(1)}$. This completes the analysis of the algorithms of theorem 7.3.1.

□

References

[1] G. W. Anderson. T-Motives. *Duke Mathematical Journal*, 53(2):457–502, June 1986.

[2] B. Anglès. On some characteristic polynomials attached to finite Drinfeld modules. *manuscripta mathematica*, 93:369–379, 1997.

[3] A. O. L. Atkin. The number of points on an elliptic curve modulo a prime (II). Available at `http://listserv.nodak.edu/archives/nmbrthry.html`, 1992.

[4] D. Ayotte, X. Caruso, A. Leudière, and J. Musleh. Drinfeld Modules in SageMath. *ACM Commun. Comput. Algebra*, 57(2):65–71, aug 2023.

[5] E. Bach, J. Driscoll, and J. Shallit. Factor Refinement. *J. Algorithms*, 15(2):199–222, September 1993.

[6] P. Berthelot. *Cohomologie Cristalline des Schémas de Caractéristique P) 0.* Lecture notes in mathematics. Springer, 1974.

[7] W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

[8] A. Bostan, F. Morain, B. Salvy, and É. Schost. Fast algorithms for computing isogenies between elliptic curves. *Math. Comput.*, 77:1755–1778, 2006.

[9] R. P. Brent, F. G. Gustavson, and D. Y. Y. Yun. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *Journal of Algorithms*, 1(3):259–295, 1980.

[10] R. P. Brent and H. T. Kung. Fast Algorithms for Manipulating Formal Power Series. *J. ACM*, 25(4):581–595, 1978.

[11] P. Bürgisser, T. Lickteig, M. Clausen, and A. Shokrollahi. *Algebraic Complexity Theory.* Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 1996.

[12] D. G. Cantor and E. L. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28:693–701, 1991.

[13] P. Caranay, M. Greenberg, and R. Scheidler. Computing modular polynomials and isogenies of rank two Drinfeld modules over finite fields. *75 Years of Mathematics of Computation*, 2020.

[14] X. Caruso and J. Le Borgne. Fast multiplication for skew polynomials. In *ISSAC'17*, pages 77–84. ACM, 2017.

[15] X. Caruso and A. Leudière. Algorithms for computing norms and characteristic polynomials on general Drinfeld modules, 2023.

[16] M. F. I. Chowdhury, C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard. Faster Algorithms for Multivariate Interpolation With Multiplicities and Simultaneous Polynomial Approximations. *IEEE Transactions on Information Theory*, 61(5):2370–2387, 2015.

[17] J.-M. Couveignes. Computing l-isogenies using the p-torsion. In Henri Cohen, editor, *Algorithmic Number Theory*, pages 59–65, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

[18] L. De Feo, C. Hugounenq, J. Plût, and É. Schost. Explicit isogenies in quadratic time in any characteristic. *LMS Journal of Computation and Mathematics*, 19(A):267–282, 2016.

[19] P. Deligne. La conjecture de Weil : I. *Publications Mathématiques de l'IHÉS*, 43:273–307, 1974.

[20] P. Deligne and D. Husemoller. Survey of Drinfel'd modules. In *Current trends in arithmetical algebraic geometry (Arcata, Calif., 1985)*, volume 67 of *Contemp. Math.*, pages 25–91. Amer. Math. Soc., Providence, RI, 1987.

[21] J. Doliskani, A. K. Narayanan, and É. Schost. Drinfeld modules with complex multiplication, Hasse invariants and factoring polynomials over finite fields. *Journal of Symbolic Computation*, 105:199–213, 2021. MICA 2016.

[22] V. G. Drinfel'd. Elliptic modules. *Matematicheskii Sbornik*, 94(23):561–593, 1974.

[23] R. Duan, H. Wu, and R. Zhou. Faster Matrix Multiplication via Asymmetric Hashing. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2129–2138, Los Alamitos, CA, USA, nov 2023. IEEE Computer Society.

[24] D. S. Dummit and R. M. Foote. *Abstract Algebra.* Wiley, 2003.

[25] B. Edixhoven. Point counting after Kedlaya. EIDMA-Stieltjes Graduate course. 01 2003.

[26] N. Elkies. Explicit isogenies. Draft, 1992.

[27] N. Elkies. Elliptic and modular curves over finite fields and related computational issues. 1997.

[28] F. Le Gall and F. Urrutia. Improved rectangular matrix multiplication using powers of the coppersmith-winograd tensor. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, page 1029–1046, USA, 2018. Society for Industrial and Applied Mathematics.

[29] S. Garai and M. Papikian. Endomorphism rings of reductions of Drinfeld modules. *Journal of Number Theory*, 212:18–39, 2020.

[30] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, New York, NY, USA, 3 edition, 2013.

[31] J. von zur Gathen and M. Giesbrecht. Constructing Normal Bases in Finite Fields. *J. Symb. Comput.*, 10:547–570, 1990.

[32] J. von zur Gathen and V. Shoup. Computing Frobenius Maps and Factoring Polynomials. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '92, page 97–105, New York, NY, USA, 1992. Association for Computing Machinery.

[33] E.-U. Gekeler. On finite Drinfeld modules. *Journal of Algebra*, 141(1):187–203, 1991.

[34] E.-U. Gekeler. Frobenius Distributions of Drinfeld Modules over Finite Fields. *Transactions of the American Mathematical Society*, 360(4):1695–1721, 2008.

[35] M. Giesbrecht. Factoring in Skew-polynomial Rings over Finite Fields. *J. Symb. Comput.*, 26(4):463 – 486, 1998.

[36] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In *ISSAC'03*, pages 135–142. ACM, 2003.

[37] D. Goss. *Basic Structures of Function Field Arithmetic*. Ergebnisse der Mathematik und Ihrer Grenzgebiete Series. Springer Berlin Heidelberg, 1997.

[38] R. Hartshorne. *Algebraic Geometry*. Graduate Texts in Mathematics. Springer, 1977.

[39] J. van der Hoeven and G. Lecerf. Composition Modulo Powers of Polynomials. In *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC '17, page 445–452, New York, NY, USA, 2017. Association for Computing Machinery.

[40] J. van der Hoeven and G. Lecerf. Fast multivariate multi-point evaluation revisited. *Journal of Complexity*, 56:101405, 2020.

[41] X. Huang and V. Y. Pan. Fast Rectangular Matrix Multiplication and Applications. *Journal of Complexity*, 14(2):257–299, 1998.

[42] A. Joux and A. K. Narayanan. Drinfeld modules may not be for isogeny based cryptography. Cryptology ePrint Archive, Paper 2019/1329, 2019. https://eprint.iacr.org/2019/1329.

[43] F. Jung. Charakteristische Polynome von Drinfeld-Moduln, 2000. Diplomarbeit, U. Saarbrücken.

[44] E. Kaltofen. Asymptotically Fast Solution of Toeplitz-like Singular Linear Systems. pages 297–304, 01 1994.

[45] E. Kaltofen and V. Pan. Processor Efficient Parallel Solution of Linear Systems over an Abstract Field. In *Proceedings of the Third Annual ACM Symposium on Parallel Algorithms and Architectures*, SPAA '91, page 180–191, New York, NY, USA, 1991. Association for Computing Machinery.

[46] E. Kaltofen and V. Shoup. Subquadratic-Time Factoring of Polynomials over Finite Fields. *Mathematics of Computation*, 67(223):1179–1197, 1998.

[47] E. Kaltofen and G. Villard. On the complexity of computing determinants. *Computational Complexity*, 13(3-4):91–130, 2004.

[48] M. Kaminski, D.G. Kirkpatrick, and N.H. Bshouty. Addition requirements for matrix and transposed matrix products. *J. Algorithms*, 9(3):354–364, 1988.

[49] K. Kedlaya. Counting Points on Hyperelliptic Curves using Monsky-Washnitzer Cohomology. *J. Ramanujan Math. Soc.*, 16, 06 2001.

[50] K. S. Kedlaya and C. Umans. Fast Polynomial Factorization and Modular Composition. *SIAM Journal on Computing*, 40(6):1767–1802, 2011.

[51] N. Kuhn and R. Pink. Finding endomorphisms of Drinfeld modules. *Journal of Number Theory*, 232:118–154, 2022. Special Issue: David Goss Memorial Issue.

[52] G. Labahn, V. Neiger, and W. Zhou. Fast, deterministic computation of the Hermite normal form and determinant of a polynomial matrix. *Journal of Complexity*, 42:44–71, 2017.

[53] R. Lercier and T. Sirvent. On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field. *Journal de Théorie des Nombres de Bordeaux*, 20(3):783–797, 2008.

[54] A. Leudière and P.-J. Spaenlehauer. Computing a group action from the class field theory of imaginary hyperelliptic function fields. *Journal of Symbolic Computation*, 125:102311, 2024.

[55] Y. Musleh. Fast Algorithms for Finding the Characteristic Polynomial of a Rank-2 Drinfeld Module. Master's thesis, 2018.

[56] Y. Musleh and É. Schost. Computing the Characteristic Polynomial of a Finite Rank Two Drinfeld Module. In *Proceedings of the 2019 on International Symposium on Symbolic and Algebraic Computation*, ISSAC '19, page 307–314, New York, NY, USA, 2019. Association for Computing Machinery.

[57] Y. Musleh and É. Schost. Computing the Characteristic Polynomial of Endomorphisms of a finite Drinfeld Module using Crystalline Cohomology. In *Proceedings of the 2023 International Symposium on Symbolic and Algebraic Computation*, ISSAC '23, page 461–469, New York, NY, USA, 2023. Association for Computing Machinery.

[58] S. N. N. Assong. *Explicit Description Of Isogeny And Isomorphism Classes Of Drinfeld Modules Of Higher Rank Over Finite Fields*. PhD thesis, Kassel, Universität Kassel, Fachbereich Mathematik und Naturwissenschaften, Institut für Mathematik, 2020.

[59] A. K. Narayanan. Polynomial factorization over finite fields by computing Euler-Poincaré characteristics of Drinfeld modules. *Finite Fields Appl.*, 54:335–365, 2018.

[60] O. Ore. Theory of Non-Commutative Polynomials. *Annals of Mathematics*, 34(3):480–508, 1933.

[61] C. H. Papadimitriou. *Computational Complexity*. Theoretical computer science. Addison-Wesley, 1994.

[62] M. Papikian. *Drinfeld Modules*. Graduate Texts in Mathematics. Springer International Publishing, 2023.

[63] I. Y. Potemine. Minimal Terminal $\mathbb{Q}$-Factorial Models of Drinfeld Coarse Moduli Schemes. *Mathematical Physics, Analysis and Geometry*, 1(2):171–191, Jun 1998.

[64] S. Puchinger and A. Wachter-Zeh. Fast operations on linearized polynomials and their applications in coding theory. *J. Symb. Comput.*, 2017.

[65] R. Schoof. Elliptic Curves Over Finite Fields and the Computation of Square Roots mod p. *Mathematics of Computation*, 44(170):483–494, 1985.

[66] J. T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *J. ACM*, 27(4):701–717, oct 1980.

[67] J.-P. Serre. Valeurs propres des endomorphismes de Frobenius. *Séminaire Bourbaki*, 16:190–204, 1973-1974.

[68] V. Shoup. Fast construction of irreducible polynomials over finite fields. *J. Symb. Comput.*, 17(5):371–391, 1994.

[69] J. H. Silverman. *The Arithmetic of Elliptic Curves.* Applications of Mathematics. Springer New York, 1986.

[70] J. H. Silverman. *Advanced Topics in the Arithmetic of Elliptic Curves.* Graduate Texts in Mathematics. Springer New York, 1994.

[71] J. H. Silverman and J. T. Tate. *Rational Points on Elliptic Curves.* Undergraduate Texts in Mathematics. Springer New York, 1994.

[72] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 10.1.0)*, 2023. `https://www.sagemath.org`.

[73] J. Vélu. Isogénies entre courbes elliptiques. *Comptes-Rendus de l'Académie des Sciences, Série I*, 273:238–241, juillet 1971.

[74] B. Wesolowski. Computing isogenies between finite Drinfeld modules. Cryptology ePrint Archive, Paper 2022/438, 2022. `https://eprint.iacr.org/2022/438`.

[75] D. H. Wiedemann. Solving Sparse Linear Equations over Finite Fields. *IEEE Trans. Inf. Theor.*, 32(1):54–62, 1986.

[76] W. Zhou, G. Labahn, and A. Storjohann. Computing Minimal Nullspace Bases. In *Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation*, ISSAC '12, page 366–373, New York, NY, USA, 2012. Association for Computing Machinery.

APPENDICES

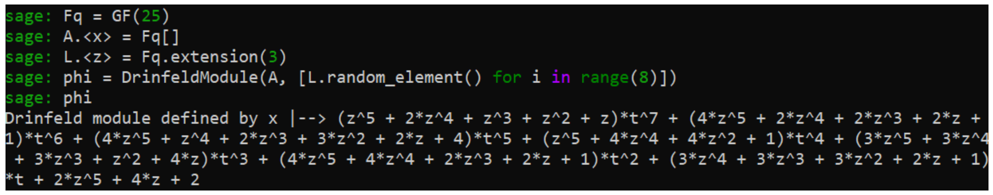
Appendix A

Software

A SageMath implementation was prepared and first presented in [4], and was developed in collaboration with David Ayotte, Xavier Caruso, and Antoine Leudière. The software provides basic features for constructing and computing with Drinfeld module objects. All methods and functionality except for `Fq_basis` and `basis` have been merged into the standard SageMath distribution. We will review some of the basic functionality here.

A.0.1 Constructing Drinfeld modules

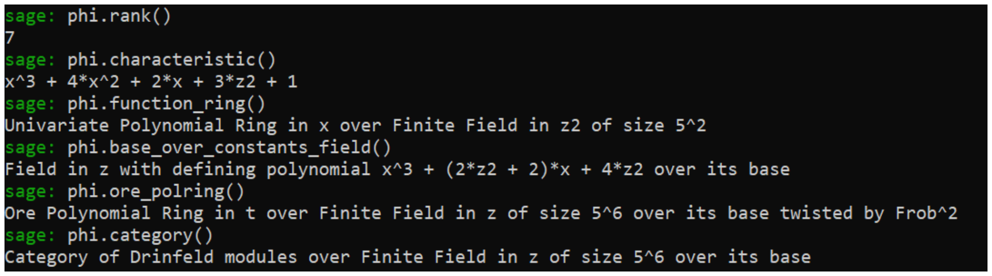
We can instantiate a Drinfeld module of rank r over a base field $\mathbb{F}_q$ and finite extension n by specifying the coefficients of ϕ_x in $\mathbb{L}$ using the `DrinfeldModule` constructor.



```
sage: Fq = GF(25)
sage: A.<x> = Fq[]
sage: L.<z> = Fq.extension(3)
sage: phi = DrinfeldModule(A, [L.random_element() for i in range(8)])
sage: phi
Drinfeld module defined by x |--> (z^5 + 2*z^4 + z^3 + z^2 + z)*t^7 + (4*z^5 + 2*z^4 + 2*z^3 + 2*z +
1)*t^6 + (4*z^5 + z^4 + 2*z^3 + 3*z^2 + 2*z + 4)*t^5 + (z^5 + 4*z^4 + 4*z^2 + 1)*t^4 + (3*z^5 + 3*z^4
 + 3*z^3 + z^2 + 4*z)*t^3 + (4*z^5 + 4*z^4 + 2*z^3 + 2*z + 1)*t^2 + (3*z^4 + 3*z^3 + 3*z^2 + 2*z + 1)
*t + 2*z^5 + 4*z + 2
```

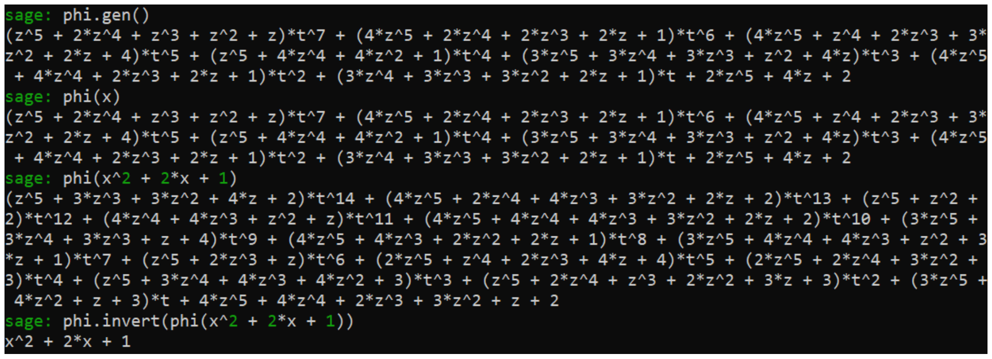
The constructor accepts two parameters: the regular function ring A and a list of coefficients $[\Delta_0, \dots, \Delta_r]$ over a ring $\mathbb{L}$ and returns a Drinfeld module $\phi : A \to \mathbb{L}\{\tau\}$ such that $\phi_x = \sum_{i=0}^{r} \Delta_i \tau^i$. Currently, only Drinfeld modules where $A = \mathbb{F}_q[x]$ for a a finite field

$\mathbb{F}_q$ are implemented. We can access basic parameters of the Drinfeld module, such as it's rank and characteristic $\mathfrak{p}$, which are inferred from the coefficients of ϕ_x. We can also access the basic algebraic objects such as $A, \mathbb{L}$, and $\mathbb{L}\{\tau\}$.



```
sage: phi.rank()
7
sage: phi.characteristic()
x^3 + 4*x^2 + 2*x + 3*z2 + 1
sage: phi.function_ring()
Univariate Polynomial Ring in x over Finite Field in z2 of size 5^2
sage: phi.base_over_constants_field()
Field in z with defining polynomial x^3 + (2*z2 + 2)*x + 4*z2 over its base
sage: phi.ore_polring()
Ore Polynomial Ring in t over Finite Field in z of size 5^6 over its base twisted by Frob^2
sage: phi.category()
Category of Drinfeld modules over Finite Field in z of size 5^6 over its base
```

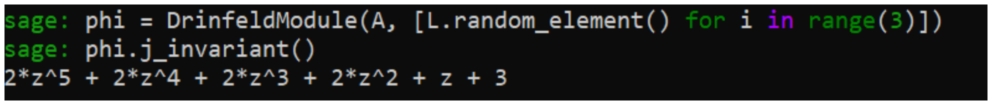
We can also access the underlying Drinfeld map $\phi : A \to \mathbb{L}\{\tau\}$. The skew polynomial ϕ_x can be accessed directly through the `gen` method. General images ϕ_a can be accessed through the operator syntax as seen below. $\phi^{-1}(s)$ can also be computed using the `invert` method.



```
sage: phi.gen()
(z^5 + 2*z^4 + z^3 + z^2 + z)*t^7 + (4*z^5 + 2*z^4 + 2*z^3 + 2*z + 1)*t^6 + (4*z^5 + z^4 + 2*z^3 + 3*
z^2 + 2*z + 4)*t^5 + (z^5 + 4*z^4 + 4*z^2 + 1)*t^4 + (3*z^5 + 3*z^4 + 3*z^3 + z^2 + 4*z)*t^3 + (4*z^5
 + 4*z^4 + 2*z^3 + 2*z + 1)*t^2 + (3*z^4 + 3*z^3 + 3*z^2 + 2*z + 1)*t + 2*z^5 + 4*z + 2
sage: phi(x)
(z^5 + 2*z^4 + z^3 + z^2 + z)*t^7 + (4*z^5 + 2*z^4 + 2*z^3 + 2*z + 1)*t^6 + (4*z^5 + z^4 + 2*z^3 + 3*
z^2 + 2*z + 4)*t^5 + (z^5 + 4*z^4 + 4*z^2 + 1)*t^4 + (3*z^5 + 3*z^4 + 3*z^3 + z^2 + 4*z)*t^3 + (4*z^5
 + 4*z^4 + 2*z^3 + 2*z + 1)*t^2 + (3*z^4 + 3*z^3 + 3*z^2 + 2*z + 1)*t + 2*z^5 + 4*z + 2
sage: phi(x^2 + 2*x + 1)
(z^5 + 3*z^3 + 3*z^2 + 4*z + 2)*t^14 + (4*z^5 + 2*z^4 + 4*z^3 + 3*z^2 + 2*z + 2)*t^13 + (z^5 + z^2 +
2)*t^12 + (4*z^4 + 4*z^3 + z^2 + z)*t^11 + (4*z^5 + 4*z^4 + 4*z^3 + 3*z^2 + 2*z + 2)*t^10 + (3*z^5 +
3*z^4 + 3*z^3 + z + 4)*t^9 + (4*z^5 + 4*z^3 + 2*z^2 + 2*z + 1)*t^8 + (3*z^5 + 4*z^4 + 4*z^3 + z^2 + 3
*z + 1)*t^7 + (z^5 + 2*z^3 + z)*t^6 + (2*z^5 + z^4 + 2*z^3 + 4*z + 4)*t^5 + (2*z^5 + 2*z^4 + 3*z^2 +
3)*t^4 + (z^5 + 3*z^4 + 4*z^3 + 4*z^2 + 3)*t^3 + (z^5 + 2*z^4 + z^3 + 2*z^2 + 3*z + 3)*t^2 + (3*z^5 +
 4*z^2 + z + 3)*t + 4*z^5 + 4*z^4 + 2*z^3 + 3*z^2 + z + 2
sage: phi.invert(phi(x^2 + 2*x + 1))
x^2 + 2*x + 1
```

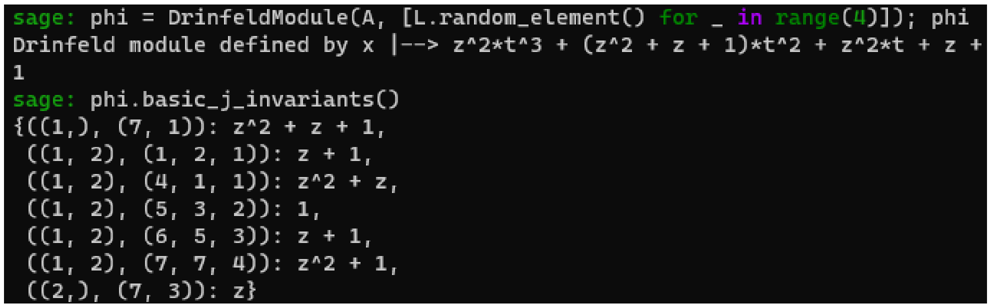
A.0.2 Invariants

In the case of rank 2 Drinfeld module, we can compute its j-invariant using the `j-invariant` method.



```
sage: phi = DrinfeldModule(A, [L.random_element() for i in range(3)])
sage: phi.j_invariant()
2*z^5 + 2*z^4 + 2*z^3 + 2*z^2 + z + 3
```

For higher rank Drinfeld modules, we can recover all of its basic j-invariants and their corresponding parameters using the `basic_j_invariants` method.

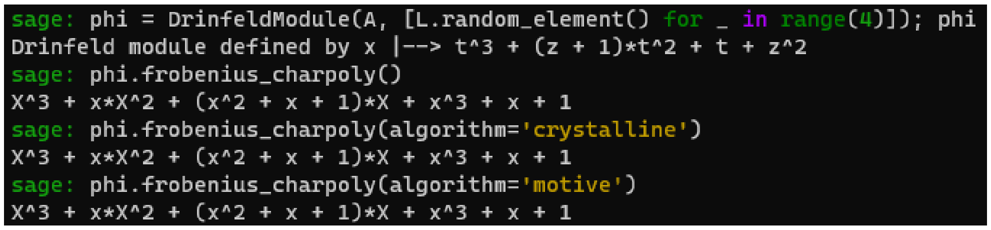


```
sage: phi = DrinfeldModule(A, [L.random_element() for _ in range(4)]); phi
Drinfeld module defined by x |--> z^2*t^3 + (z^2 + z + 1)*t^2 + z^2*t + z +
1
sage: phi.basic_j_invariants()
{((1,), (7, 1)): z^2 + z + 1,
 ((1, 2), (1, 2, 1)): z + 1,
 ((1, 2), (4, 1, 1)): z^2 + z,
 ((1, 2), (5, 3, 2)): 1,
 ((1, 2), (6, 5, 3)): z + 1,
 ((1, 2), (7, 7, 4)): z^2 + 1,
 ((2,), (7, 3)): z}
```

Recalling the context of definition 4.3.1, the parameters are returned in the format

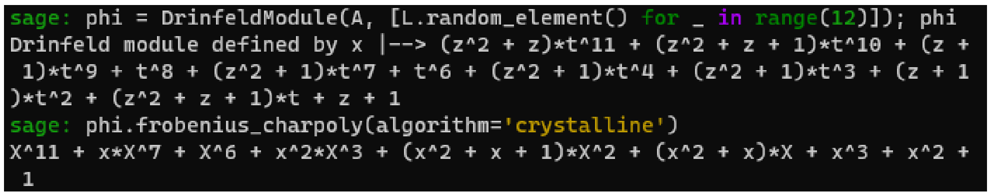
$$((k_1, \dots, k_h), (\delta_1, \dots, \delta_h, \delta_r)) : J_{k_1,\dots,k_h}^{\delta_1,\dots,\delta_h}(\phi)$$

We can compute the characteristic polynomial of the Frobenius endomorphism using the `frobenius_charpoly` method. We may pass an optional `algorithm` parameter to this method to specify the procedure used.



```
sage: phi = DrinfeldModule(A, [L.random_element() for _ in range(4)]); phi
Drinfeld module defined by x |--> t^3 + (z + 1)*t^2 + t + z^2
sage: phi.frobenius_charpoly()
X^3 + x*X^2 + (x^2 + x + 1)*X + x^3 + x + 1
sage: phi.frobenius_charpoly(algorithm='crystalline')
X^3 + x*X^2 + (x^2 + x + 1)*X + x^3 + x + 1
sage: phi.frobenius_charpoly(algorithm='motive')
X^3 + x*X^2 + (x^2 + x + 1)*X + x^3 + x + 1
```

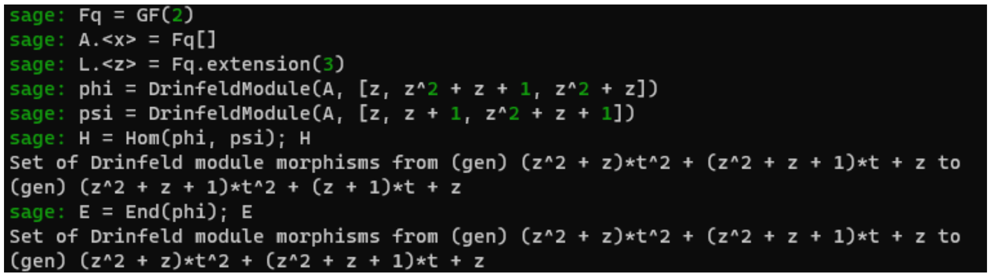
It is also possible to use this method with Drinfeld modules of rank larger than 2.



```
sage: phi = DrinfeldModule(A, [L.random_element() for _ in range(12)]); phi
Drinfeld module defined by x |--> (z^2 + z)*t^11 + (z^2 + z + 1)*t^10 + (z +
 1)*t^9 + t^8 + (z^2 + 1)*t^7 + t^6 + (z^2 + 1)*t^4 + (z^2 + 1)*t^3 + (z + 1
)*t^2 + (z^2 + z + 1)*t + z + 1
sage: phi.frobenius_charpoly(algorithm='crystalline')
X^11 + x*X^7 + X^6 + x^2*X^3 + (x^2 + x + 1)*X^2 + (x^2 + x)*X + x^3 + x^2 +
 1
```

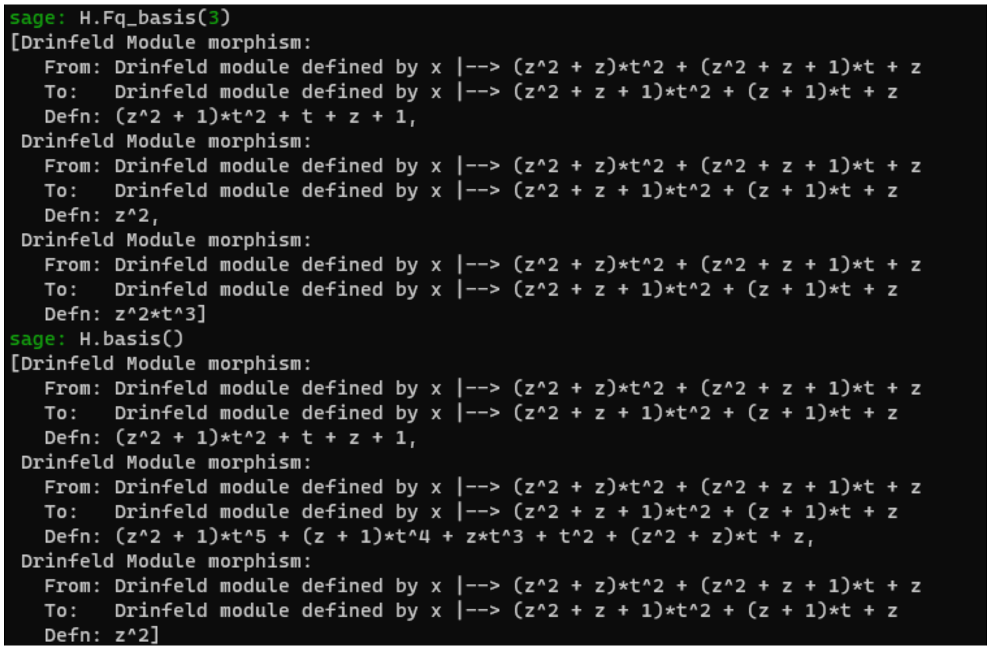
A.0.3 Morphisms

We can also instantiate the set of homomorphisms between Drinfeld modules ϕ, ψ or endomorphisms on ϕ using SageMath's built-in `Hom` and `End` constructors.



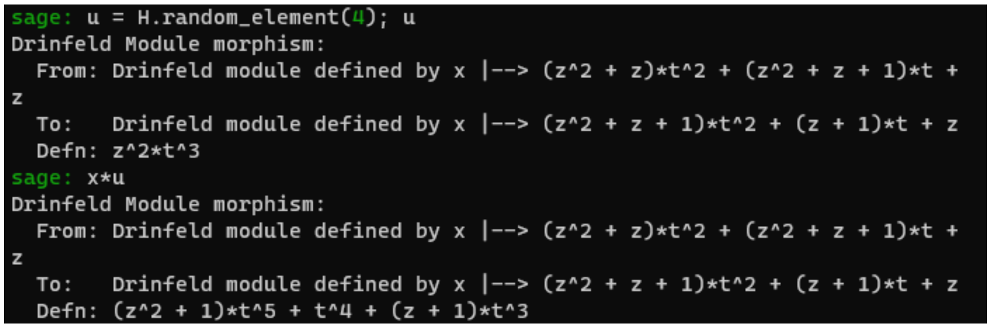
```
sage: Fq = GF(2)
sage: A.<x> = Fq[]
sage: L.<z> = Fq.extension(3)
sage: phi = DrinfeldModule(A, [z, z^2 + z + 1, z^2 + z])
sage: psi = DrinfeldModule(A, [z, z + 1, z^2 + z + 1])
sage: H = Hom(phi, psi); H
Set of Drinfeld module morphisms from (gen) (z^2 + z)*t^2 + (z^2 + z + 1)*t + z to
(gen) (z^2 + z + 1)*t^2 + (z + 1)*t + z
sage: E = End(phi); E
Set of Drinfeld module morphisms from (gen) (z^2 + z)*t^2 + (z^2 + z + 1)*t + z to
(gen) (z^2 + z)*t^2 + (z^2 + z + 1)*t + z
```

Currently, it is possible to compute an explicit basis for the space of morphisms over $\mathbb{F}_q$ of degree at most d using the `Fq_basis` method, or over $\mathbb{F}_q[\tau^n]$ using the `basis` method. These methods are currently awaiting review for inclusion into SageMath.



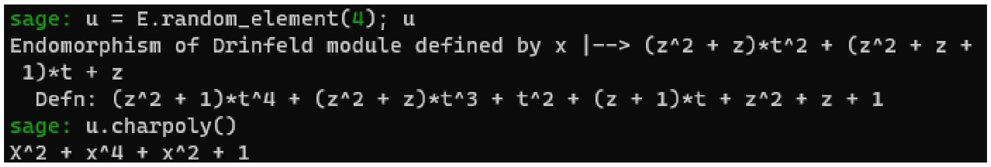
```
sage: H.Fq_basis(3)
[Drinfeld Module morphism:
   From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t + z
   To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
   Defn: (z^2 + 1)*t^2 + t + z + 1,
 Drinfeld Module morphism:
   From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t + z
   To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
   Defn: z^2,
 Drinfeld Module morphism:
   From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t + z
   To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
   Defn: z^2*t^3]
sage: H.basis()
[Drinfeld Module morphism:
   From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t + z
   To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
   Defn: (z^2 + 1)*t^2 + t + z + 1,
 Drinfeld Module morphism:
   From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t + z
   To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
   Defn: (z^2 + 1)*t^5 + (z + 1)*t^4 + z*t^3 + t^2 + (z^2 + z)*t + z,
 Drinfeld Module morphism:
   From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t + z
   To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
   Defn: z^2]
```

Recall the action of $\mathbb{F}_q[x]$ on $\mathrm{Hom}(\phi, \psi)$ given by $a * u = u\phi_a$ for $a \in \mathbb{F}_q[x]$ and $u \in \mathrm{Hom}(\phi, \psi)$. This action can be applied using the multiplication operator in SageMath.



```
sage: u = H.random_element(4); u
Drinfeld Module morphism:
  From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t +
z
  To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
  Defn: z^2*t^3
sage: x*u
Drinfeld Module morphism:
  From: Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z + 1)*t +
z
  To:   Drinfeld module defined by x |--> (z^2 + z + 1)*t^2 + (z + 1)*t + z
  Defn: (z^2 + 1)*t^5 + t^4 + (z + 1)*t^3
```

The characteristic polynomial of an endomorphism can be computed using the `charpoly` method.



```
sage: u = E.random_element(4); u
Endomorphism of Drinfeld module defined by x |--> (z^2 + z)*t^2 + (z^2 + z +
 1)*t + z
  Defn: (z^2 + 1)*t^4 + (z^2 + z)*t^3 + t^2 + (z + 1)*t + z^2 + z + 1
sage: u.charpoly()
X^2 + x^4 + x^2 + 1
```