Fast Algorithms for Finding the Characteristic Polynomial of a Rank-2 Drinfeld Module

by

Yossef Musleh

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2018

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This thesis introduces a new Monte Carlo randomized algorithm for computing the characteristic polynomial of a rank-2 Drinfeld module. We also introduce a deterministic algorithm that uses some ideas seen in Schoof's algorithm for counting points on elliptic curves over finite fields. Both approaches are a significant improvement over the current literature.

Acknowledgements

I would like to thank all the people who made this thesis possible. In particular, I would like to thank my supervisor Eric Schost, whose exceptional effort made this thesis possible. I would also like to thank the readers for their helpful suggestions, as well as my family and friends for their support.

Dedication

This is dedicated to my mother and late father.

Table of Contents

List of Tables

List of Figures

Chapter 1

Motivation

Elliptic curves are one of the most well-studied Algebro-geometric objects. Their deep importance to Number theory is emphasized by a connection to several famous theorems and conjectures, including Fermat's Last theorem and the Birch and Swinnerton-Dyer conjecture. Among the most ground breaking results is Schoof's algorithm for counting the number of points on an elliptic curve over a finite field [26]. Along with later improvements due to Elkies and Atkin, Schoof's algorithm provides an approach for point counting which is practical for use in cryptography. Elliptic curve cryptography using the Schoof-Atkin-Elkies algorithm is currently the most widely used implementation of pre-quantum public key cryptosystem.

Drinfeld modules were originally introduced in [7] who used them to partially prove the Langlands conjecture for $GL(2, F)$ when F is an algebraic function field. This proof was later completed using *shtukas*, a further generalization of Drinfeld modules. The original name of "elliptic module" emphasizes the fundamental connection between elliptic curves and Drinfeld modules, with the latter intended as a function field analogue of the former. In this vein, significant scholarship has been devoted to producing Drinfeld analogues of elliptic curve constructions, including cryptographic protocols which are highly insecure [25]. The primary motivation of this thesis is to extend this scholarship by providing efficient algorithms for performing computations on Drinfeld modules, and in particular studying techniques for computing the *characteristic polynomial* of a Drinfeld module, which characterizes equivalence classes of Drinfeld modules up to isogeny and performs a critical role in Schoof's algorithm for elliptic curves.

The main results of this thesis are two new algorithms for computing the characteristic polynomial of a rank-2 Drinfeld module. The best algorithm given in the literature, due to Gekeler, runs in $O(n^3 \log^2 q)$ field operations. In contrast, our new randomized algorithm takes $O(n^2 \log n \log\log n \log q)$, while our deterministic approach uses approximately $O(n^{2.6258} \log n + n^2 \log n \log\log n \log q)$ operations. Also included is a new divide-and-conquer algorithm for evaluating the characteristic map of a Drinfeld module which runs in $O(n^2 rd \log(rd) \log q)$ field operations.

Chapter 2

Preliminaries

2.1 Basic Algorithms and Notation

There will be core algorithms and notation that we will make use of throughout this work, but will largely treat as black boxes. We will assume the reader is familiar with the algebraic concepts of groups, rings, and fields. In determining the complexity of the algorithms presented, two computational models will be used:

- an algebraic model for algorithms over a ring or field, in which the standard ring operations of addition and multiplication can be performed at unit cost.
- a boolean model which counts bit complexity of all operations.

The algebraic model will typically be preferred, however the Kedlaya-Umans algorithm for modular composition does not always admit an algebraic algorithm. Algorithms utilizing modular composition will have their complexity given in both an algebraic and boolean model.

Operation 1 (Matrix Multiplication)**.** *Let R be a ring, and let M be an $m \times n$ matrix, and N an $n \times \ell$ matrix, both with entries contained in R. Compute $M \times N$.*

We will let ω denote an exponent such that two $n \times n$ matrices can be multiplied using $O(n^\omega)$ ring operations in any ring R. The current best known bound is $\omega \approx 2.3728$ [9]. Similarly, we let ω_2 denote the exponent of $n \times n$ and $n \times n^2$ multiplication, with the best known bound of $\omega_2 < 3.2516$ [9], and naively $\omega_2 \leq \omega + 1$.

Operation 2 (Polynomial Multiplication)**.** *Let R be a commutative ring, and let f, g be polynomials in $R[T]$. Compute $f \cdot g$.*

We will let $M(n)$ denote the the complexity of polynomial multiplication when f and g are both of degree at most n. For rings containing sufficiently many primitive

roots of unity, methods based on the Fast Fourier Transform achieve operation counts of $M(n) \in O(n \log n)$ [10, Chapter 8]. For general rings, the best known ring operation count is $M(n) \in O(n \log n \log\log n)$ [4].

We may interpret a polynomial $f = \sum_{i=0}^{m} c_i T^i$ as a vector whose i^{th} entry is c_i, and may similarly interpret such a vector as a polynomial. We will also have a standard notion of interpreting a list of polynomials $\{f_1, \ldots, f_n\} \subset R[T]$ of degree at most m as the matrix $M \in R^{(m+1)\times n}$ whose i^{th} column is the vector f_i. We will also implicitly assume that, for a prime power q, any finite field written as $\mathbb{F}_{q^m}$ is implemented as a polynomial ring over $\mathbb{F}_q$ modulo an irreducible of degree m.

Definition 1. *Let f, g be polynomials of degrees d_1, d_2 respectively, with coefficients in a commutative ring R, and roots $\alpha_1, \ldots, \alpha_{d_1}$, $\beta_1, \ldots, \beta_{d_2}$ in some extension of R. The* ***resultant*** *of f and g is defined to be*

$$\operatorname{res}(f, g) := f_{d_1}^{d_2} g_{d_2}^{d_1} \prod_{i=1}^{d_1} \prod_{j=1}^{d_2} (\alpha_i - \beta_j).$$

2.2 Modular Operations

Operation 3 (Modular Composition)**.** *Let R be a ring, and $f, g, h \in R[T]$ with h monic. Compute*

$$f(g(T)) \bmod h(T).$$

Fix $\deg(f), \deg(g), \deg(h) \leq d$. Naively using polynomial multiplication to substitute g into f yields a runtime of $O(dM(d))$ operations in R. The classical algorithm for modular composition due to Brent and Kung [2] solves modular composition using $O(d^{\frac{\omega+1}{2}})$ operations in F. A slight improvement to this by Huang and Pan gives a count of $O(d^{\frac{\omega_2}{2}})$[14], which for current estimates of ω_2 is approximately $O(n^{1.6258})$. More recent work due to Kedlaya and Umans, utilizing a reduction to multipoint evaluation and an FFT approach, gives an operation count of $O(d^{1+\delta} \log^{1+o(1)} |R|)$ for any $\delta > 0$ when R is a finite ring of the form $\mathbb{Z}/r\mathbb{Z}[T]/E(T)$ for some positive integer r and some polynomial $E(T) \in \mathbb{Z}/r\mathbb{Z}[T]$ and containing at least $d^{1+\delta}$ elements whose differences are units [19, Theorem 7.1]. Moreover, they show that this runtime is optimal up to lower order terms. Since the algorithm involves lifting the ring to characteristic 0, the complexity is determined in a bit operation model rather than an algebraic one.

Operation 4 (Automorphism Projection)**.** *Given a field L, automorphism σ of L fixing a subfield K with $m = [L : K]$, and a K-linear map $u : L \to K$, for any $\alpha \in L$ and positive integer k, compute $u(\sigma^i(\alpha))$ for all $0 \leq i \leq k$.*

For a prime power q, when $K = \mathbb{F}_q$, $L = \mathbb{F}_{q^m}$, and $\sigma : \alpha \mapsto \alpha^q$ is the order q Frobenius, we can take $k \leq m-1$ and an algebraic baby-step/giant-step algorithm due to Kaltofen and Shoup solves the automorphism projection problem in $O(m^{\omega_2/2+(1-\beta)(\omega-1)/2+o(1)} + m^{1+\beta+o(1)} \log q)$ field operations in $\mathbb{F}_q$ for any $0 \leq \beta \leq 1$. For the current best bounds of ω, ω_2, taking $\beta = 0.6258$ yields a complexity of $O(m^{1.6258+o(1)} \log q)$. Kedlaya and Umans extend their approach for modular composition to produce an algorithm to solve automorphism projection in $O(d^{1+\delta} \log |L|)$ bit operations when L is finite and σ is the order q Frobenius.

2.3 Automorphisms and Normal Bases

We again let L be a field, K a subfield of L, and let $\sigma : L \to L$ be a K-automorphism.

Definition 2. *A basis $\{b_0, \ldots, b_{m-1}\}$ for L as a vector space over K is **normal** with respect to σ if $\sigma(b_i) = b_{i-1}$ for $i = 0, \ldots, m-1$, with indices taken modulo m.*

Operation 5 (Normal Basis Construction)**.** *Find an element $b \in L$ such that $\{b, \sigma(b), \ldots, \sigma^{m-1}(b)\}$ is a basis for L/K.*

In the finite field case, with $K = \mathbb{F}_q$, $L = \mathbb{F}_{q^m}$, $\sigma : \alpha \mapsto \alpha^q$, von zur Gathen and Giesbrecht [28] give a randomized algorithm that runs in $O(m^2 \log q)$ $\mathbb{F}_q$. Kaltofen and Shoup in [18] give an improved algorithm, again randomized, that determines a normal element in $O(m^{\omega_2/2+(1-\beta)(\omega-1)/2+o(1)} + m^{1+\beta+o(1)} \log q)$ $\mathbb{F}_q$ operations for any $0 \leq \beta \leq 1$. Kedlaya and Umans in [19] produce an overall runtime for selecting a normal basis, and converting to and from the standard basis for $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$, of $O(m^{\omega_2/2} \log^{1+o(1)} q + m^{1+o(1)} \log^{2+o(1)} q)$ bit operations. This is achieved by utilizing the algorithm due to Kaltofen and Shoup, which depends on algorithms for automorphism projection and evaluation, in conjunction with their own asymptotically optimal algorithm for modular composition and modular power projection. This yields a complexity of roughly $O(m^{1.6258+o(1)} \log q)$ $\mathbb{F}_q$ operations for Kaltofen-Shoup when $\beta = 0.6258$, and $O(m^{1.6258} \log^{1+o(1)} q + m^{1+o(1)} \log^{2+o(1)} q)$ bit operations for Kedlaya-Umans.

2.3.1 Linear Recurring Sequences

Definition 3. *A sequence $\{a_i\}_{i=0}^{\infty}$ with entries in a field L is said to be a **linear recurring** sequence if there exists $c_0, c_1, \ldots, c_{d-1}$ in L such that*

$$a_{d+j} + c_{d-1}a_{d+j-1} + c_{d-2}a_{d+j-2} + \ldots + c_0 a_j = 0 \qquad \forall j \geq 0$$

The polynomial $x^d + \sum_{i=0}^{d-1} c_i x^i$ is called a characteristic polynomial of $\{a_i\}_{i=0}^{\infty}$. The minimal polynomial of the sequence is the unique monic polynomial $\sum_{i=0}^{d} k_i x^i$ of smallest degree d such that $\sum_{i=0}^{d} k_i a_{j+i} = 0$ for all $j \geq 0$.

Operation 6. *Given a linear recurring sequence* $\{a_i\}_{i=0}^{\infty}$*, compute its minimal polynomial.*

If the linear recurring sequence $\{a_i\}_{i=0}^{\infty}$ is known to have a minimal polynomial of degree at most d, the Berlekamp-Massey algorithm [21] determines the minimal polynomial given $2d$ entries in $O(d^2)$ field operations over any field.

Chapter 3

Fast Multiplication of Skew Polynomials

As part of the general view towards providing efficient algorithms for basic operations on Drinfeld modules, efficient techniques for arithmetic on skew polynomial rings become necessary. We begin with a field $K = \mathbb{F}_q$ and an irreducible commutative polynomial $\mathfrak{p} \in \mathbb{F}_q[x]$ of degree m. Let $L = \mathbb{F}_q[x]/\mathfrak{p} = \mathbb{F}_{q^m}$ be a field extension of degree m, and let σ be a K-automorphism of L. Then the ring of *skew polynomials* $L[X, \sigma]$ consists of polynomials in a new variable X with coefficients in L together with the commutation relation $Xa = \sigma(a)X$ for $a \in L$.

Example 1. *Let $q = 2$, $\mathfrak{p} = x^2 + x + 1$. Then $L = \{0, 1, x, x + 1\}$, $\sigma : x \mapsto x^2$, and we have the following defining relations for $L[X, \sigma]$:*

$$X \cdot x = (x + 1) \cdot X$$
$$X \cdot (x + 1) = x \cdot X.$$

The goal of this section is to examine efficient algorithms to solve the following problem.

Operation 7 (Skew Polynomial Multiplication)**.** *Let $a, b \in L[X, \sigma]$. Compute $a \cdot b$.*

An algorithm due to Giesbrecht [13] for multiplying skew polynomials of degree at most d in $L[X, \sigma]$ has a runtime of $O(d^2 M(m) \log q + dmM(m) \log m \log q)$. In the following section we will provide two more recent algorithms.

3.1 Puchinger and Wachter-Zeh's Algorithm

The first algorithm for skew polynomial multiplication we will present is due to Puchinger and Wachter-Zeh [24], and is valid when σ is the order q Frobenius map. We provide a

more detailed analysis than in [24], particularly in light of the fact that, in their original work, the authors assume that K-automorphisms of L can be evaluated in constant time, whereas here we will drop this assumption.

Theorem 1. *Let $a, b \in \mathbb{F}_{q^m}[X, \sigma]$ be skew polynomials of degree at most d. The product $a \cdot b$ can be computed in*

- $O(d^{\omega_2/2}M(m) + d^{3/2}m^{\omega_2/2}))$ $\mathbb{F}_q$ *operations in the algebraic model using Brent-Kung Modular composition*
- $O(d^{\omega_2/2}M(m) + d^{3/2}m^{1+\delta}\log^{1+o(1)} q)$ *bit operations using the Kedlaya-Umans algorithm for modular composition.*

Proof. Let $d^* = \lceil\sqrt{d+1}\rceil$ We write a and b as

$$a = \sum_{i=0}^{d} a_i X^i$$

$$b = \sum_{i=0}^{d} b_i X^i$$

for $a_i, b_i \in \mathbb{F}_{q^m}$. Now define

$$\begin{aligned} a^{(i)} &= \sum_{j=0}^{d^*-1} a_{id^*+j} X^{id^*+j} \\ a &= \sum_{i=0}^{d^*-1}\sum_{j=0}^{d^*-1} a_{id^*+j} X^{id^*+j} \\ &= \sum_{i=0}^{d^*-1} a^{(i)}. \end{aligned}$$

Let then

$$c^{(i)} = a^{(i)} \cdot b, \text{ for } i = 0, \ldots, d^* - 1.$$

Then we can rewrite $a \cdot b$ as

$$c = \sum_{i=0}^{d^*-1} a^{(i)} \cdot b = \sum_{i=0}^{d^*-1} c^{(i)}.$$

For $i = 0, \ldots, d^* - 1$, each $c^{(i)}$ can be expanded as

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

For $h = 0, \ldots, d + d^* - 1$, let

$$
c_h^{(i)} = \sum_{j=0}^{h} a_{id^*+j} \sigma^{id^*+j}(b_{h-j}).
$$

Then

$$
c = \sum_{i=0}^{d^*-1} \sum_{h=0}^{d+d^*-1} c_h^{(i)}
$$

and

$$
\sigma^{-id^*}(c_h^{(i)}) = \sum_{j=0}^{h} \sigma^{-id^*}(a_{id^*+j}) \sigma^j(b_{h-j}).
$$

Let A be a $d^* \times d^*$ and B, C be $d^* \times (d + d^*)$ matrices such that

$$
\begin{aligned}
A_{i,j} &= \sigma^{-id^*}(a_{id^*+j}) \\
B_{i,k} &= \sigma^i(b_{k-i}) \\
C_{i,k} &= \sigma^{-id^*}(c_k^{(i)})
\end{aligned}
\tag{1}
$$

for $0 \leq i, j < d^*$, $0 \leq k < d + d^*$. By (1) we have $C = A \cdot B$. This leads to the following algorithm for computing c:

1. Build matrices A, B as given in (1)
2. Compute the matrix product $A \cdot B$
3. Determine $c_j^{(i)}$ from the entries of C and compute c

In determining the complexity of step 1, the following two approaches may be used to compute the entries of A, B:

(a) Compute matrices for σ, σ^{-1} as K-linear maps on L and compute matrix powers

(b) Use "polynomial representations" of both σ, σ^{-1} and fast modular composition.

The notion of a "polynomial representation" of the Frobenius map, which is credited to Kaltofen in [29], proceeds as follows: first compute $\alpha := x^q \bmod \mathfrak{p}$, and then exploit the observation that, for any element $g(x) \in \mathbb{F}_{q^m}[x]/\mathfrak{p}$, $g(x^q) = g(x)^q$. This allows evaluation of the Frobenius map by computing the modular composition of g with α.

In either case, we begin by representing the automorphism $\sigma : x \mapsto x^q$ on $\mathbb{F}_{q^m}$ as α, a polynomial of degree at most $m-1$, which can be done in $O(M(m)\log q)$ $\mathbb{F}_q$-operations.

Using the matricial approach, computing the matrix representation M_σ requires evaluating σ on the standard polynomial basis $\{1, x, \ldots, x^{m-1}\}$ for $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$, which can be done by computing $\alpha^2, \ldots, \alpha^{m-1}$ in time $O(M(m)m)$, and $M_{\sigma^{-1}}$ can be computed from M_σ in $O(m^\omega)$.

To determine the entries of A we compute $M_{\sigma^{-id^*}}$ for $1 \le i \le d^*$. Computation of the matrix power $M_{\sigma^{-d^*}}$ costs $O(m^\omega \log d)$, and computing $M_{\sigma^{-id^*}}$ for all $1 \le i \le d^*$ takes $O(d^* m^\omega)$. With d possible values for a_{id^*+j}, evaluation costs $O(m^2 d)$ for all entries. To determine the entries of B we determine M_{σ^i} for $i \le d + d^*$ with a total cost $O(dm^\omega)$ $\mathbb{F}_q$-operations using matrix multiplication, and evaluation takes $O(d^{3/2}m^2)$ since there are $d^*(d+d^*)$ entries of the form $\sigma^i(b_{k-i})$ to compute. The overall runtime of step 1 using the matricial approach is therefore $O(\log(q)M(m) + dm^\omega + d^{3/2}m^2)$.

We may instead compute σ^i by composing α with itself. Each individual composition takes $O(m^{\omega_2/2})$ field operations using Brent-Kung, or $m^{1+\delta}\log^{1+o(1)} q$ bit operations using Kedlaya-Umans. The polynomial representation of σ^{-1} can be determined by computing $\sigma^{q^{m-1}}$, which can be done in $O(\log m)$ compositions. Computing the entries of A of the form $\sigma^{-id^*}(a_{id^*+j})$ requires $O(\log(d))$ compositions to compute σ^{-d^*}, and a further $O(d)$ compositions to compute each of $\sigma^{-id^*}(a_{id^*+j})$ for all choices of $0 \le i, j \le d^*$. Moreover, to determine B, we use up to $O(d^{3/2})$ compositions to compute all entries of the form $\sigma^i(b_{k-j})$ for $i \le k$, $i \le d^*$, $k < d+d^*$. This gives step 1 a runtime of $O(d^{3/2}m^{\omega_2/2} + M(m)\log m)$ field operations using Brent-Kung, or $O(d^{3/2}m^{1+\delta} + M(m)\log m)$ bit operations using Kedlaya-Umans. The best overall runtime of the procedure is then $O(d^{\omega_2/2}M(m) + d^{3/2}m^{1+\delta})$ $\mathbb{F}_q$-operations using Kedlaya-Umans modular composition when $p \le d^{o(1)}$.

Step two takes the time of multiplying an $d^* \times d^*$ matrix with an $d^* \times (d+d^*)$ matrix, which is no more than $O((d^*)^{\omega_2}M(m)) = O(d^{\omega_2/2}M(m))$. Computing $c_j^{(i)}$ requires computing and evaluating $\sigma^{id^*}(C_{i,k})$ for $i < d^*$, $k < d+d^*$ which adds $O(d^{3/2}m^2)$ operations using the matricial approach or $O(d^{3/2}m)$ using the polynomial representation.

□

Under the current bounds for ω_2 and $M(m)$, the Brent-Kung approach has a complexity of $O(d^{1.6258}m\log m\log\log m + d^{1.5}m^{1.6258})$ $\mathbb{F}_q$ operations, while the Kedlaya-Umans

Algorithm 1 Puchinger-Wachter-Zeh (Matrix Multiplication)

procedure SkewMultiplication
  Input A prime power q, integer m, skew polynomials a, b
  Output $a \cdot b$
  $d \leftarrow \mathbf{Max}(\mathbf{Degree}(a), \mathbf{Degree}(b))$
  $d^* \leftarrow \lceil \sqrt{d+1} \rceil$
  $M_\sigma \leftarrow \mathbf{Matrix}(1, x^q, x^{2q}, \ldots, x^{(m-1)q})$
  $M_{\sigma^{-1}} \leftarrow M_\sigma^{-1}$
  $M_{\sigma^{-d^*}} \leftarrow M_{\sigma^{-1}}^{d^*}$
  $M_{\sigma^{d^*}} \leftarrow M_\sigma^{d^*}$
  for $i = 2$ **to** $d^* - 1$ **do**
    $M_{\sigma^{-id^*}} \leftarrow M_{\sigma^{-(i-1)d^*}} M_{\sigma^{-d^*}}$
  for $i = 2$ **to** $d + d^* - 1$ **do**
    $M_{\sigma^i} \leftarrow M_{\sigma^{i-1}} M_\sigma$
  for $i = 0$ **to** $d^* - 1$ **do**
    for $j = 0$ **to** $d^* - 1$ **do**
      $A[i, j] \leftarrow M_{\sigma^{-id^*}}(a_{id^*+j})$
  for $i = 0$ **to** $d^* - 1$ **do**
    for $k = 0$ **to** $d + d^* - 1$ **do**
      $B[i, k] \leftarrow M_{\sigma^i}(b_{k-i})$
  $C \leftarrow \mathbf{MatrixMultiply}(A, B)$
  $c \leftarrow 0$
  for $i = 0$ **to** $d^* - 1$ **do**
    for $i = 0$ **to** $d + d^* - 1$ **do**
      $c \leftarrow c + M_{\sigma^{id^*}}(C[i, k])$
  return c

Algorithm 2 Puchinger-Wachter-Zeh (Modular Composition)

procedure SkewMultiplication
  Input A prime power q, integer m, skew polynomials a, b
  Output $a \cdot b$
  ModularCompose$(\alpha, i) := \overbrace{\alpha \circ \alpha \circ \ldots \circ \alpha}^{i \text{ times}}$
  $d \leftarrow$ **Max**(**Degree**(a), **Degree**(b))
  $\alpha \leftarrow x^q \bmod \mathfrak{p}$
  $d^* \leftarrow \lceil \sqrt{d+1} \rceil$
  $\alpha_{-1} \leftarrow$ **ModularCompose**$(\alpha, m-1)$
  $\alpha_{-d^*} \leftarrow$ **ModularCompose**(α_{-1}, d^*)
  for $i = 2$ **to** $d^* - 1$ **do**
    $\alpha_{\sigma^{-id^*}} \leftarrow \alpha_{\sigma^{-(i-1)d^*}} \circ \alpha_{\sigma^{-d^*}}$
  for $i = 2$ **to** $d + d^* - 1$ **do**
    $\alpha_{\sigma^i} \leftarrow \alpha_{\sigma^{i-1}} \circ \alpha_{\sigma}$
  for $i = 0$ **to** $d^* - 1$ **do**
    for $j = 0$ **to** $d^* - 1$ **do**
      $A[i, j] \leftarrow \alpha_{\sigma^{-id^*}} \circ a_{id^*+j}$
  for $i = 0$ **to** $d^* - 1$ **do**
    for $k = 0$ **to** $d + d^* - 1$ **do**
      $B[i, k] \leftarrow \alpha_{\sigma^i} \circ b_{k-i}$
  $C \leftarrow$ **MatrixMultiply**(A, B)
  $c \leftarrow 0$
  for $i = 0$ **to** $d^* - 1$ **do**
    for $i = 0$ **to** $d + d^* - 1$ **do**
      $c \leftarrow c + \alpha_{\sigma^{id^*}} \circ C[i, k]$
  return c

variant has a complexity of $O(d^{1.6258} m \log m \log\log m + d^{1.5} m^{1+\delta} \log^{1+o(1)} q)$ bit operations. The pseudocode for the Puchinger-Wachter-Zeh algorithm is given in figures 1 and 3.

The main limiting factor is computing the evaluations of the σ^i. The next algorithm we will look at will attempt to eliminate this by exploiting a normal basis for σ.

3.2 Fast Multiplication from Evaluation on Normal Bases

A more recent algorithm for multiplication of skew polynomials is given by Caruso and Le Borgne in [5], which again requires that $L = \mathbb{F}_{q^m}$. Their main results are contained in the following two theorems:

Theorem 2. *Let $a, b \in L[X, \sigma]$ such that $\deg(a) + \deg b \leq d < m$. Then there is an algorithm that can compute $a \cdot b$ in $O(d^{\omega-2} m^2 + m^{\omega_2/2+(1-\beta)(\omega-1)/2+o(1)} + m^{1+\beta+o(1)} \log q)$ K operations for any constant $0 \leq \beta \leq 1$ [5].*

Theorem 3. *Let $a, b \in L[X, \sigma]$ such that $\deg(a), \deg(b) \geq m$. Then there is a probabilistic algorithm that can compute $a \cdot b$ in $O(dm^{\omega-1} + m^{\omega_2/2+(1-\beta)(\omega-1)/2+o(1)} + m^{1+\beta+o(1)} \log q)$ K operations for any constant $0 \leq \beta \leq 1$ with likelihood at least $\frac{1}{2}$. [5].*

The first lemma, given in [5, lemma 1.4], establishes a natural correspondence between elements of $L[X, \sigma]$ and $\mathrm{End}_K(L)$.

Lemma 1. *The map*

$$\begin{aligned} \chi : L[X, \sigma] &\to \mathrm{End}_K(L) \\ X &\mapsto \sigma \\ \sum_i a_i X^i &\mapsto \sum_i a_i \sigma^i \end{aligned}$$

induces an isomorphism $\chi : L[X, \sigma]/(X^m - 1) \to \mathrm{End}_K(L)$.

The next lemma, which is [5, proposition 1.6], allows us to exploit multiplication on *commuting* polynomials to evaluate the automorphisms efficiently on normal bases:

Lemma 2. *Suppose $A(X) = \sum_i a_i X^i \in L[X, \sigma]$, and let $\bar{A}(T) = \sum_i a_i T^i$ be the polynomial with identical coefficients in a commuting variable T. Let $B(T) = \sum_{i=0}^{m-1} b_i T^i$, $C(T) = \sum_i \chi(A)(b_i) T^i$. Then:*

$$C(T) = \bar{A}(T) B(T) \pmod{T^m - 1}.$$

The proofs for Theorems 2 and 3 rely largely on a more general version of the following result, which is sufficient for multiplying two skew polynomials of total degree at most $m-1$. We present a more detailed version of the proof that appears in [5, corollary 1.7], to account for the cost of constructing and converting to and from a normal basis.

Theorem 4. *Let $a, b \in L[X,\sigma]/(X^m-1)$. The product $a \cdot b$ can be computed in $O(m^\omega + m^{\omega_2/2+(1-\beta)(\omega-1)/2+o(1)} + m^{1+\beta+o(1)} \log q)$ $\mathbb{F}_q$ operations.*

Proof. Select a normal basis $\{\beta_0, \ldots, \beta_{m-1}\}$ of L over $\mathbb{F}_q$ using the Kaltofen-Shoup algorithm seen in 2.3, and let Ω represent the change of basis matrix from a chosen standard basis to the the normal one. Assuming the standard basis is the usual power basis $\{1, x, x^2, \ldots\}$, Ω can be constructed from the coefficients of the previously computed normal basis in $O(m^2)$ $\mathbb{F}_q$-operations. Moreover let $B(T) = \sum_{i=0}^{m-1} \beta_i T^i$. Compute $a'(T) = \overline{a}(T)B(T)$, $b'(T) = \overline{b}(T)B(T)$ which can be done in $O(M(m)^2)$ $\mathbb{F}_q$-operations using commutative polynomial multiplication. From Lemma 2, $a'(T) = \sum_{i=0}^{m-1} \chi(a)(\beta_i)T^i$, and so we may construct the matrices of $\chi(a), \chi(b)$, where the domain uses the normal basis and the codomain has the standard one, by extracting the coefficients of a' and b' in $O(m^2)$ operations. Then $\chi(ab) = \chi(a)\Omega\chi(b)$ which can be computed in $O(m^\omega)$ $\mathbb{F}_q$-operations, and the column entries give the coefficients $\chi(ab)(\beta_i)$ which allows us to compute $c'(T) = \sum_{i=0}^{m-1} \chi(ab)(\beta_i)T^i$ in $O(m^2)$ operations. Using Lemma 2 again, we have that $c'(T)B^{-1}(T) = \overline{ab}(T)$, which takes at most $O(M(m))$, and the result is obtained.

□

Algorithm 3 Caruso-Le Borgne

procedure SkewMultiplication
  Input A prime power q, integer m, $a = \sum_{i=0}^{m-1} a_i X^i$, $b = \sum_{i=0}^{m-1} b_i X^i \in \mathbb{F}_{q^m}[X,\sigma]$
  Output $a \cdot b$
  Generate a normal basis $\{\beta_0, \ldots, \beta_{m-1}\}$
  $\Omega \leftarrow$ **Matrix**$(\beta_0, \ldots, \beta_{m-1})$
  $B \leftarrow \sum_{i=0}^{m-1} \beta_i T^i$
  $\overline{a} \leftarrow \sum_{i=0}^{m-1} a_i T^i$
  $\overline{b} \leftarrow \sum_{i=0}^{m-1} b_i T^i$
  $a' \leftarrow \overline{a}B$
  $b' \leftarrow \overline{b}B$
  $M_{\chi(a)} \leftarrow$ **Matrix**$(\chi(a)(\beta_0), \ldots, \chi(a)(\beta_{m-1}))$
  $M_{\chi(b)} \leftarrow$ **Matrix**$(\chi(b)(\beta_0), \ldots, \chi(b)(\beta_{m-1}))$
  $C \leftarrow$ **MatrixMultiply**$(M_{\chi(a)}, \Omega, M_{\chi(b)})$
  for $i = 0$ **to** $m-1$ **do**
    $C_i \leftarrow i^{th}$ column of C
  $c' \leftarrow \sum_{i=0}^{m-1} C_i T^i$
  $c \leftarrow c'(T)B^{-1}(T)$
  return $c(X)$

3.3 Summary and Comparison

The bottleneck in the Puchinger and Wachter-Zeh approach is the computation of powers and evaluation of the automorphism σ. Representing σ and σ^{-1} as polynomials and utilizing fast modular composition to compute the σ^i eliminates an m^ω term over the matricial approach guaranteeing a run time at worst quadratic in m. The Caruso-Le Borgne manages to evade this difficulty completely by relying on a normal basis relative to σ, which in the case where σ is the Frobenius map we can guarantee the existence of with natural assumptions, and algorithms to compute these bases are well studied.

The Caruso-Le Borgne algorithm's runtimes of $sm^{\omega-1}$ (for large degree) and $s^{\omega-2}m^2$ (for small degree) offer a lower total degree than Brent-Kung's $s^{3/2}m^{\frac{\omega}{2}}$. The Puchinger-Wachter-Zeh approach offers an asymptotic advantage over the Caruso-Le Borgne algorithm in the low degree case when we roughly have $s^{3/2} \leq \sqrt{m}$.

Chapter 4

Elliptic Curves

The primary objective of this thesis is to develop a function field analog of techniques used in counting rational points on elliptic curves. To that end, we first present classical concepts and results pertaining to elliptic curves. In particular we will describe Schoof's algorithm, which together with improvements due to Atkin and Elkies is one of the fastest methods available.

4.1 Mathematical Background

We will give a basic introduction to some of the core concepts of classical algebraic geometry. For further exposition, see [8].

Definition 4. *Let L be a field. An (affine irreducible) **variety** is a set $V \subset L^n$ such that there exist polynomials $f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n) \in L[x_1, \ldots, x_n]$ such that for all $v \in V$ we have that $f_i(v) = 0$ for $1 \leq i \leq m$. We may replace L^n with n-dimensional projective space $\mathbb{P}^n$ to obtain the notion of a **projective variety**, in which case we also require that each f_i is a homogeneous polynomial.*

Example 2. *Let L be a field and $n = 2$. The polynomial $f(x, y, z) = x^3 + axz^2 + bz^3 - y^2z$ is irreducible for any $a, b \in L$ and therefore the set $V = \{(x, y, z) : f(x, y, z) = 0\}$ is a projective variety over $\mathbb{P}^2$*

For the purposes of this thesis we will consider "projective curves" with domain $L^2 \cup \{\infty\}$; the plane together with a "point at infinity". Over this domain we can represent $f(x, y, z)$ from the preceding example as $f(x, y) = x^3 + ax^2 + b - y^2$.

Definition 5. *An **abelian variety** is a projective variety whose points form an abelian group whose group operation and element inversion can be defined by regular maps.*

Definition 6. *An **elliptic curve** E is a smooth abelian variety over $L^2 \cup \{\infty\}$ defined by the affine equation*

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

For $a, b \in L$ such that $4a^3 + 27b^2 \neq 0$.

Addition on E can be defined as follows:

$$\infty + \infty = \infty$$
$$(x, y) + \infty = (x, y)$$

For $x, y \in L$, let:

$$\mu = \frac{3x^2 + a}{2y}$$

and then we have:

$$2(x, y) = (\mu^2 - 2x, 3\mu x - \mu^3 - y)$$

.

For any pair of points $(x_1, y_1), (x_2, y_2)$ with $x_1 \neq x_2$, define:

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}.$$

Then we have:

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

When L is finite, we let $|E|$ denote the number of points on the curve in $L^2 \cup \{\infty\}$.

Operation 8. *Given an elliptic curve E over a finite field, compute $|E|$.*

4.2 Schoof's Algorithm for Elliptic Curves

Schoof's algorithm for counting the number of points on an elliptic curve is based on a result due to Hasse:

Theorem 5 (Hasse)**.** *Let E be an elliptic curve over $\mathbb{F}_q$. Then*

$$||E| - q - 1| \leq 2\sqrt{q}$$

Letting $h = |E| - q - 1$, by Hasse's theorem, it is sufficient to compute h modulo any $m \geq 4\sqrt{q}$. The approach used by Schoof computes $h \mod p$ for a number of primes p_i such that $\prod_i p_i \geq 4\sqrt{q}$ and uses the Chinese Remainder theorem to then reconstruct h. We now let σ denote the order q Frobenius map. Moreover, let $\overline{E}$ be the curve with the same defining equation of E but whose entries now come from $\overline{\mathbb{F}}_q$, the algebraic closure of $\mathbb{F}_q$. We then have the following theorem.

Theorem 6 (Hasse's Theorem for Elliptic Curves)**.** *Let σ be the order q Frobenius endomorphism defined over $\overline{\mathbb{F}}_q$. Then σ has the following characteristic equation over $\overline{E}$:*

$$X^2 + hX + q = 0$$

For any $a \in \mathbb{Z}$, define $a(x,y) := \overbrace{(x,y) + \ldots + (x,y)}^{a \text{ times}}$, then Hasse's theorem tells us that, for any point $(x,y) \in \overline{E}$, we have that:

$$\sigma^2(x,y) + q(x,y) = -h\sigma(x,y)$$

.

Rather than attempt to solve for h directly, we restrict to the p-torsion subgroups $E_p = \{(x,y) \in \overline{E} : p(x,y) = \infty\}$ for a set of primes $p \in P$, and compute an integer h_p such that the restricted morphisms satisfy the equation

$$\mu_p(x,y) := \sigma_p^2(x,y) + q_p(x,y) = -h_p\sigma_\ell(x,y) \text{ for } (x,y) \in E_\ell$$

where $q_\ell \equiv q \pmod{p}$, $h_p \equiv h \pmod{p}$.

In order to compute the restricted morphism μ_p, we construct a class of polynomials in $\mathbb{F}_q[x,y]$ whose roots are exactly the elements of E_p inductively 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. *The points $(x,y) \in \overline{E}$ satisfying $\psi_i(x,y) = 0$ are exactly the set E_i.*

To determine $h_p \pmod{p}$, we let $\mu_p(x,y) = (\mu_p^x(x,y), \mu_p^y(x,y))$, where $\mu_p^x(x,y), \mu_p^y(x,y)$ are rational functions over $\mathbb{F}_{||}[x,y]/(y^2 - x^3 - ax - b, \psi_p)$ computing the x-coordinate and y-coordinate of $\mu_p(x,y)$ respectively. We may then solve for an h_p such that $(\mu_p^x(x,y), \mu_p^y(x,y))$ $= -h_p\sigma(x,y)$. Having computed h_p for each $p \in P$, we can use the Chinese Remainder Theorem to solve for an integer h such that $h = h_p \mod p$ for all $p \in P$, and by Hasse's theorem, $|E| = h + q + 1$.

The runtime of Schoof's original algorithm in [26] is $O(\log^{5+o(1)} q)$.

Algorithm 4 Schoof's Algorithm for Point Counting on Elliptic Curves

procedure EllipticPointCount
  Input A prime power q and an elliptic curve E represented by $y^2 = x^3 - ax + b$
  Output $|E|$
  Fix a set of primes $P \leftarrow \{p : \gcd(p, q) = 1\}$ and $\prod_{p \in P} p > 4\sqrt{q}$
  for each $p \in P$ **do**
    $\psi_p \leftarrow$ **DivisionPolynomial**(p)
    $\mu_p(x, y) \leftarrow \sigma^2(x, y) + q(x, y) \bmod y^2 - x^3 - ax - b, \psi_p$
    solve $\mu_p(x, y) = -h_p\sigma(x, y)$ for $h_p \bmod p$
  $h \leftarrow$ **CRTSolve**$(\{h_p \mod p\}_{p \in P})$
  Return $h + q + 1$

4.3 The Hasse Invariant

An alternate approach to point counting utilizes the *Hasse invariant* of an elliptic curve

Definition 7. *The Hasse invariant $H_q(E)$ of an elliptic curve E over the field $\mathbb{F}_q$ with characteristic p and defined by the equation $y^2 = f(x) = x^3 + ax + b$ is the coefficient of x^{q-1} in $f^{(q-1)/2}$.*

Theorem 7. *For any prime power $q = p^e$ and elliptic curve E over $\mathbb{F}_q$ we have:*

$$|E| = 1 - H_q(E) \bmod p.$$

If $q = p$, this uniquely determines $H_q(E)$.

In the prime field case, this leads to an alternative algorithm for point counting on elliptic curves that runs in $O(p)$ $\mathbb{F}_p$ operations by simply expanding the expression for $f^{(q-1)/2}$. More involved algorithms reduce this to $O(\sqrt{p})$ operations [1].

Chapter 5

Drinfeld Modules

The primary goal of this thesis is to translate machinery for point counting from the classical elliptic case to the function field analog. To this end, we now introduce some theoretical machinery required to introduce the generic definition of a Drinfeld module.

5.1 Mathematical Background

Definition 8. *The (finite)* ***transcendence degree*** *of a field L over a subfield K is the largest cardinality n of a set $\{x_1, \dots x_n\}$ contained in L such that there is no non-zero polynomial f in n variables with coefficients in K with $f(x_1, \dots, x_n) = 0$. If such a set can be found for all $n > 0$ then we say that the transcendence degree of L/K is infinite.*

Example 3. *Let K be any field, $T_1, \dots, T_n$ a set of n indeterminates and $L = K(T_1, \dots, T_n)$. Then L has transcendence degree n over K.*

Definition 9. *A* ***function field*** *L over a field K is a field extension of K with transcendence degree $n \geq 1$*

Given a field extension L/K, any element $\ell \in L$ corresponds to a linear operator $\hat{\ell} : L \to L$ defined by $\hat{\ell} : a \mapsto \ell a$. Let M_ℓ be the matrix for $\hat{\ell}$, with entries in K.

Definition 10. *Let L/K be a field extension. For any $\ell \in L$, define:*

$$N_{L/K}(\ell) := \det(M_\ell)$$
$$Tr_{L/K}(\ell) := \text{trace}(M_\ell)$$

which are referred to as the ***norm*** *and* ***trace*** *of ℓ respectively.*

5.1.1 Valuations

Definition 11. *A **valuation** on a field F into an ordered abelian group G is a map $v : F \to G \cup \{\infty\}$ satisfying the following conditions:*

1. *$v(a) = \infty$ if and only if $a = 0$*
2. *$v(ab) = v(a) + v(b)$*
3. *$v(a + b) \geq \min(v(a), v(b))$*

Example 4. *Let $F = \mathbb{Q}$. For any prime p we can define the p-adic valuation $\nu_p : \mathbb{Q} \to \mathbb{Z} \cup \{\infty\}$ as follows:*

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

Two valuations $v_1 : F \to G_1$, $v_2 : F \to G_2$ are *equivalent* if there is an order-preserving group isomorphism $\varphi : G_2 \to G_1$ such that $v_1(a) = \varphi(v_2(a))$.

Example 5. *Recalling the example of the p-adic valuations on $\mathbb{Q}$, if p_1, p_2 are two distinct primes, then $\nu_{p_1}(p_2) = 0 = \varphi(\nu_{p_2}(p_2)) = \varphi(1)$, which implies φ is the trivial map. So all p-adic valuations are non-equivalent.*

Definition 12. *A **place** of a field F is an equivalence class of valuations.*

In the typical number field setting, one can construct the the *finite* places of a number field F as follows: consider the ring of integers Z of F and let $\mathfrak{p}$ be a maximal prime ideal. We define $v : Z \to \mathbb{Z}$ by setting $v(a)$ to be the largest n such that $a \in \mathfrak{p}^n$, and then extend v to F as done in the p-adic case form the preceding example. This same construction can be repeated for $F(T)$, over an underlying field F, by replacing the ring of integers with $F[T]$.

5.2 Drinfeld Modules

Definition 13. *Let F be a function field with a place ∞, and A a sub-ring of F of elements regular at every point except possibly ∞ containing the field $\mathbb{F}_q$ for a prime power q. Consider a map $\gamma : A \to L$ and set $\sigma : a \mapsto a^q$. Then a **Drinfeld A-module** over L is an injective morphism $\phi : A \to L[X, \sigma]$ such that*

$$\phi(a) = \gamma(a) + a_1 X + \ldots a_r X^r$$

With the further requirement that $r \geq 1$ and $a_r \neq 0$ for at least one $a \in A$.

There is a simpler construction when L is an extension of a finite field $\mathbb{F}_q$, say $L = \mathbb{F}_q[T]/\mathfrak{f} = \mathbb{F}_{q^n}$ for some monic irreducible $\mathfrak{f} \in \mathbb{F}_q[T]$ of degree n. Here we have $A = \mathbb{F}_q[T]$, and we fix an inclusion $\gamma : A \to L$ where $\ker(\gamma)$ is a prime ideal of A generated by a monic irreducible polynomial $\mathfrak{p}$ of degree d referred to as the *A-Characteristic* of L. Then $K = A/\mathfrak{p}$ and $\gamma(a)$ represents the canonical mapping of $a \in A$ into L. Let $m := [L : K]$. Finally, let $\sigma : L \to L$ be the order q Frobenius map. Then a Drinfeld module is a ring homomorphism $\phi : A \to L[X, \sigma]$ such that

$$\phi(a) = \gamma(a) + a_1 X + \ldots a_r X^r$$

and $r \geq 1$ and $a_r \neq 0$ for at least one $a \in A$. For the remainder of this thesis, this will be the definition we will make use of. If $a = T$, a Drinfeld Module is completely specified by the parameters $(q, \mathfrak{f}, \mathfrak{p}, a_1, \ldots, a_r)$ or as $(q, \mathfrak{f}, \mathfrak{p}, \phi_T)$, though we will frequently drop the references to q,$\mathfrak{f}$, and $\mathfrak{p}$. Since each of these parameters can be specified with coefficients in $\mathbb{F}_q$, all cost analysis will be done in the algebraic model using $\mathbb{F}_q$ operations unless specified otherwise.

Example 6. *Let $q = 2$, $A = \mathbb{F}_2[T]$, $f = T^2 + T + 1$, and $L = A/f = \mathbb{F}_4$. Then let $\gamma : A \to L$ be the quotient map by f and define the Drinfeld Module $\phi : A \to L$ as*

$$\phi_T = T + X + X^2$$

We will often write ϕ_a in place of $\phi(a)$. Since ϕ is a ring homomorphism, we have $\phi_{ab} = \phi_a \phi_b$, and so the Drinfeld module is determined entirely by ϕ_T; the degree of ϕ_T is referred to as the *rank* of the Drinfeld Module.

There is a map ι of elements of $L[X, \sigma]$ into $End_{\mathbb{F}_q}[L]$ given by:

$$\iota : a_0 + a_1 X + a_2 X^2 + \ldots \mapsto a_0 I + a_1 \sigma + a_2 \sigma^2 + \ldots$$

where $I : L \to L$ is the identity operator. This mapping is a ring homomorphism, and given a Drinfeld module ϕ we will interpret elements ϕ_a as operators $L \to L$ under this association for all $a \in A$.

We will restrict our consideration to rank-2 Drinfeld modules, which are widely viewed in the literature as a direct function field analogue of elliptic curves [11]. Letting $\phi_T = \gamma(T) + gX + \Delta X^2$, we can represent any rank-2 Drinfeld module with the pair (g, Δ). From [12], we have the following theorem.

Theorem 8. *Suppose ϕ is a rank-2 Drinfeld module over L, a degree n extension of $\mathbb{F}_q$, and suppose σ is the order q Frobenius map. Then there is a polynomial $T^2 - aT + b \in A[T]$ such that the absolute Frobenius operator $\tau = \sigma^n : L \to L$ satisfies the characteristic equation*

$$\tau^2 - \phi_a \tau + \phi_b = 0 \tag{5.1}$$

as an element of $L[X, \sigma]$ under ι; that is, $(X^n)^2 - \phi_a X^n + \phi_b = 0$ in $L[X, \sigma]$. The coefficients a and b are referred to as the *Frobenius trace* and *Frobenius norm* respectively.

5.3 An Algorithm for the Characteristic Map

One of the most basic operations that can be performed on a Drinfeld module is the computation of $\phi_a \in L[X, \sigma]$ for an arbitrary element $a = \sum_{i=0}^{d} a_i T^i \in A$.

Operation 9 (Evaluating the Characteristic Map)**.** *Given a Drinfeld Module $\phi : A \to L$ and an element $a \in A$, compute ϕ_a.*

Let θ be the exponent such that two skew-polynomials in $\mathbb{F}_{q^m}[X, \sigma]$ of degree at most e can be multiplied in $O(e^\theta)$ $\mathbb{F}_q$ operations, dropping dependence on all other parameters. Since ϕ is a ring homomorphism, we have that:

$$\phi : \sum_{i=0}^{d} a_i T^i \mapsto \sum_{i=0}^{d} a_i \phi_T^i$$

which leads to a naive algorithm for computing ϕ_a by computing ϕ_T^i via skew-polynomial multiplication for all $2 \leq i \leq d$, which works for any rank r Drinfeld Module. This involves at least $d/2$ multiplications of skew-polynomials of degree at least $rd/2$, and therefore has a complexity of $\Omega(d^{\theta+1})$.

Theorem 9. *There is an algorithm that evaluates the characteristic map of any finite Drinfeld module $(q, \mathfrak{f}, \mathfrak{p}, \phi_T)$ of rank r at an element $a \in \mathbb{F}_q[T]$ of degree at most d in $O(n^2 rd \log(rd) \log q)$ $\mathbb{F}_q$ operations.*

Proof. Given an element $a \in \mathbb{F}_q[T]$, without loss of generality we may assume $\deg(a) = d = 2^e$ for some integer e. Then we may factor a as

$$a = b + T^{d/2} c$$

with $\deg(b), \deg(c) \leq d/2$. We then have:

$$\phi_a = \phi_b + \phi_{T^{d/2}} \phi_c$$

where each of ϕ_b, $\phi_{T^{d/2}}$, and ϕ_c have degree at most $\frac{rd}{2}$ in X. This suggests a divide and conquer algorithm which recursively computes ϕ_b, ϕ_c and $\phi_{T^{d/2}} = \phi_{T^{d/4}} \phi_{T^{d/4}}$ when $e \geq 2$, with runtime $O(t(n, dr))$ $\mathbb{F}_q$ operations for some t depending on n, d, q, and r.

Suppressing the dependence on n and q, set $\overline{d} = rd$ and $\hat{t}(\overline{d}) = t(n, d, q, r)$. Using the Caruso-Le Borgne algorithm for skew multiplication twice to compute $\phi_{T^{d/2}}$ and $\phi_{T^{d/2}}\phi_c$, which from theorems 2 and 3 has worst case runtime $O(\overline{d}n^2 \log q)$ for any value of $\overline{d}$, n, and q, to recombine the subproblems, we obtain the following recurrence:

$$\hat{t}(\overline{d}) = 2\hat{t}(\overline{d}/2) + O(\overline{d}).$$

Therefore $\hat{t} \in O(\overline{d} \log \overline{d})$, and $O(t(n, d, r)) \in O(m^2 rd \log(rd) \log q)$.

□

Algorithm 5 Evaluating the Drinfeld Characteristic Map

procedure CHARMAPEVAL
  Background A Drinfeld Module $(q, \mathfrak{f}, \mathfrak{p}, \phi_T)$
  Input An element $a \in A$
  Output ϕ_a
  $a := \sum_{i=0}^{d} a_i T^i$
  if $\deg(a) \leq 1$ **then return** $a_0 + a_1\phi_T$
  $b \leftarrow \sum_{i=0}^{\lfloor d/2 \rfloor} a_i T^i$
  $c \leftarrow \sum_{i=0}^{\lfloor d/2 \rfloor} a_{i+\lceil d/2 \rceil} T^i$
  Memoize $\phi_{\lfloor d/2 \rfloor} \leftarrow$ **SkewMultiplication**$(\phi_{\lfloor \lfloor d/2 \rfloor /2 \rfloor}, \phi_{\lfloor \lfloor d/2 \rfloor /2 \rfloor})$
  if d is odd **then**
    $\phi_{\lceil d/2 \rceil} \leftarrow$ **SkewMultiplication**$(\phi_{\lfloor \lfloor d/2 \rfloor /2 \rfloor}, \phi_T)$
  else
    $\phi_{\lceil d/2 \rceil} \leftarrow \phi_{\lfloor d/2 \rfloor}$
  $\phi_b \leftarrow$ **CharMapEval**(b)
  $\phi_c \leftarrow$ **CharMapEval**(c)
  return $\phi_b +$ **SkewMultiplication**$(\phi_{\lceil d/2 \rceil}, \phi_c)$

5.4 Previous Algorithms for Computing the Characteristic Polynomial of the Frobenius Map

We now state the central problem of this thesis

Operation 10 (Computing the Characteristic Polynomial)**.** *Given a rank-2 Drinfeld module $\phi = (g, \Delta)$, compute its Frobenius Trace and Norm.*

5.4.1 Gekeler's Algorithm

Determining the Frobenius norm is done using the following theorem from [12]:

Theorem 10. *The Frobenius norm b of a rank-2 Drinfeld Module (g, Δ) is given by:*

$$b = (-1)^n N_{\mathbb{F}_{q^n}/\mathbb{F}_q}(\Delta)^{-1}\mathfrak{p}^m$$

Furthermore, the above expression can be determined in $O(M(n)\log n)$ $\mathbb{F}_q$ operations.

Proof. We will provide only the cost analysis, which is new. Note $\mathfrak{p}^m$ is a degree n polynomial and can compute $\mathfrak{p}^m$ in time $O(M(n)\log m)$ $\mathbb{F}_q$ operations. Moreover $N_{L/\mathbb{F}_q}(\Delta) = \pm\mathrm{res}(\Delta, \mathfrak{f})$ [23], which can be computed in $O(M(n)\log n)$ $\mathbb{F}_q$ operations [23]. □

Gekeler in [12] gives a general algorithm that determines the Frobenius trace for any rank-2 Drinfeld module. We present that algorithm now, together with a new cost analysis which is not provided in the original paper.

Theorem 11. *The Frobenius trace of a rank-2 Drinfeld Module $\phi = (g, \Delta)$ over L can be determined in*

- $O(n^3 \log^2 q)$ $\mathbb{F}_q$ *operations*
- $O(n^{3+\epsilon} \log^{1+o(1)} q)$ *bit operations for any $\epsilon > 0$.*

Proof. From [12] we obtain that $\deg(a) \leq \frac{n}{2}$. Set :

$$\begin{aligned} a &= \sum_{i \leq n/2} a_i T^i \\ \mathfrak{p}^m &= \sum_{i \leq n} p_i T^i \\ \phi_{T^i} &= \sum_{j \leq 2i} f_{i,j} X^j \end{aligned}$$

With a_i,p_i, and $f_{i,j} \in \mathbb{F}_{q^n}$ for all choices of i, j, and $f_{i,0} = \gamma(T^i)$, $f_{1,1} = g$, $f_{1,2} = \Delta$. Using theorem 10, we can compute ϕ_b in $O(M(n)\log n)$ $\mathbb{F}_q$ operations, and by theorem 8 we have:

$$\sigma^{2n} + \sum_{i \leq n/2} a_i \phi_{T^i} \sigma^n + \phi_b = 0$$

which gives

$$\sigma^{2n} + \sum_{i \leq n/2} a_i \sum_{j \leq 2i} f_{i,j}\sigma^{j+n} + (-1)^n N_{L/\mathbb{F}_q}(\Delta)^{-1} \sum_{i \leq n} p_i \sum_{j \leq 2i} f_{i,j}\sigma^j = 0.$$

Factoring out σ^{2n}, this gives a system of $n+1$ equations

$$-\sum_{i\leq n/2} a_i f_{i,j-n} + (-1)^n N_{L/\mathbb{F}_q}(\Delta)^{-1} \sum_{\lfloor j/2\rfloor \leq i\leq n} p_i f_{i,j} = 0 \text{ for } j < 2n$$

and

$$-\sum_{i\leq n/2} a_i f_{i,n} + (-1)^n N_{L/\mathbb{F}_q}(\Delta)^{-1} p_n f_{n,2n} = -1$$

with p_i already computed while determining the Frobenius norm, and $f_{i,j}$ determined using a method that is to be discussed later. From known values, we can compute

$$\begin{aligned}\alpha_j &= (-1)^n N_{L/\mathbb{F}_q}(\Delta)^{-1} \sum_{\lfloor (j+n)/2\rfloor \leq i\leq n} p_i f_{i,j+n} \text{ for } 0 \leq j < n \\ \alpha_n &= 1 + (-1)^n N_{L/\mathbb{F}_q}(\Delta)^{-1} p_n f_{n,2n}\end{aligned}$$

Since $\lfloor \frac{n}{2}\rfloor = \lfloor \frac{n+1}{2}\rfloor$ when n is even, we have that the equations are redundant when $j \in \{n+2i-1 : 1 \leq i \leq \frac{n}{2}\}$ for even n, while for odd n this occurs when $j \in \{n+2i+2 : 0 \leq i \leq \frac{n-1}{2}\}$. Eliminating redundancies and noting that $f_{i,j} = 0$ whenever $i < 2j$ leaves the following upper triangular system of $\lfloor \frac{n}{2}\rfloor + 1$ equations

$$\begin{bmatrix} f_{0,0} & f_{1,0} & \cdots & f_{\lfloor n/2\rfloor,0} \\ 0 & f_{1,2} & \cdots & f_{\lfloor n/2\rfloor,2} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f_{\lfloor n/2\rfloor,n} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{\lfloor n/2\rfloor} \end{bmatrix} = \begin{bmatrix} \alpha_0 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} \tag{5.2}$$

whose diagonal entries are $f_{i,2i}$, which are the coefficients of the leading terms of ϕ_{T^i}, and therefore are of the form $f_{i,2i} = \Delta^e \neq 0$ for some exponent e and for $0 \leq i \leq \lfloor \frac{n}{2}\rfloor$ since $\Delta \neq 0$. The upper triangular system can be solved in $O(n^2)$ $\mathbb{F}_{q^n}$ operations, and therefore $O(n^2 M(n))$ $\mathbb{F}_q$ operations, by solving for $a_{\lfloor n/2\rfloor}$ using $f_{\lfloor n/2\rfloor,n} a_{\lfloor n/2\rfloor} = \alpha_n$ and recursively solving for a_i using the solution for $a_{i+1}, \ldots, a_{\lfloor n/2\rfloor}$. However, determining the system requires computing $f_{i,j}$ for $i \leq n/2$, $j \leq n$. Techniques for skew polynomial multiplication offer a naive approach, however an improvement can be obtained by determining a recurrence:

$$\begin{aligned}\phi_{T^{i+1}} = \phi_T \phi_{T^i} &= (\gamma(T) + gX + \Delta X^2) \sum_{j\leq 2i} f_{i,j} X^j \\ &= \sum_{j\leq 2i} \gamma(T) f_{i,j} X^j + \sum_{j\leq 2i} g f_{i,j}^q X^{j+1} + \sum_{j\leq 2i} \Delta f_{i,j}^{q^2} X^{j+2}\end{aligned}$$

So the $f_{i,j}$ satisfy the recurrence

$$f_{i+1,j} = \gamma(T) f_{i,j} + g f_{i,j-1}^q + \Delta f_{i,j-2}^{q^2}$$

With initial conditions $f_{0,0} = 1$, $f_{1,0} = \gamma(T)$, $f_{1,1} = g$, and $f_{1,2=\Delta}$. Evaluating one instance of the recurrence involves three applications of the Frobenius map, which is the dominating step, and takes $O(M(n) \log q)$ $\mathbb{F}_q$ operations. There are $O(n^2)$ choices of i and j on which we want to evaluate this recurrence and so we obtain a worst case runtime of $O(M(n)n^2 \log q)$ field operations to determine all $f_{i,j}$. Using Kedlaya-Umans modular composition, however, gives a bit operation count of $O(in^{1+\epsilon \log^{1+o(1)} q})$, and computes all $f_{i,j}$ in time $O(n^{3+\epsilon} \log^{1+o(1)} q)$ for any constant $\epsilon > 0$. The cubic complexity in n is due to the need to solve $O(n)$ recurrences to determine all $f_{i,j}$, and therefore can not be repaired in an elementary manner.

□

Algorithm 6 Gekeler's Algorithm for the Frobenius Trace

procedure FROBENIUSTRACE
  Input A rank-2 Drinfeld Module (g, Δ)
  Output The Frobenius Trace
  $f_{0,0} \leftarrow 1$
  $f_{1,0} \leftarrow \gamma(T)$
  $f_{1,1} \leftarrow g$
  $f_{1,2} \leftarrow \Delta$
  for $i = 2$ **to** $\lfloor \frac{n}{2} \rfloor$ **do**
    for $j = 1$ **to** $\lfloor \frac{n}{2} \rfloor$ **do**
      $f_{i,j} \leftarrow (T \bmod \mathfrak{p}) \cdot f_{i-1,j-1} + g \cdot f_{i-1,j-2}^q + \Delta \cdot f_{i-1,j-3}^{q^2}$
  for $i = 0$ **to** $\lfloor \frac{n}{2} \rfloor$ **do**
    for $j = 0$ **to** $\lfloor \frac{n}{2} \rfloor$ **do**
      $A[j, i] \leftarrow f_{i,2j}$
  for $j = 0$ **to** $\lfloor \frac{n}{2} \rfloor$ **do**
    $\alpha[j] \leftarrow (-1)^n N_{L/\mathbb{F}_q}(\Delta)^{-1} \sum_{\lfloor (2j+n)/2 \rfloor \leq i \leq n} p_i f_{i,2j+n}$
  Solve $Ax = \alpha$
  Return x

5.4.2 The Case $L = K$

Gekeler in [12] gives a much simpler algorithm in the case where $L = K$. In analogy with the elliptic case, we may define the Hasse Invariant h_ϕ for a rank-2 Drinfeld module.

Definition 14. *Let $(q, \mathfrak{f}, \mathfrak{p}, g, \delta)$ be a rank-2 drinfeld module with $\mathfrak{f} = 0$. We define the **Hasse invariant** h_ϕ of ϕ to be the coefficient of X^n in $\phi_\mathfrak{p}$.*

Theorem 12. *Let (g, Δ) be a rank-2 Drinfeld module, $L = \mathbb{F}_q[T]/\mathfrak{p}$, $[L : \mathbb{F}_q] = n$. We let g_i for $i \leq n$ be the sequence defined by $g_0 = 1$, $g_1 = g$ and*

$$g_{k+1} = g^{q^k} g_k - (T^{q^k} - T)\Delta^{q^{k-1}} g_{k-1}.$$

Then $\gamma(a) = h_\phi = g_n$

Using fast modular composition and techniques for solving linear recurrences with polynomial coefficients as given in [6], we obtain a runtime for solving the recurrence for g_n of $O\big(n^{(1-\beta)(\omega-1)/2+(\omega+1)/2} + M(n^{1+\beta}\log qn)\big)$ $\mathbb{F}_q$ operations for any positive constant $\beta < 1$. Again making use of Kedlaya-Umans modular composition, all of g^{q^i}, Δ^{q^i}, and $T^{q^i} - T$ for $i < n$ can be computed in $O(n^{2+\epsilon}\log^{1+o(1)} q)$ bit operations, and therefore g_n can be computed with the same asymptotic cost.

5.4.3 Narayanan's Algorithm

A first randomized approach based on computing minimal polynomials of sequences due to Narayanan [22, 3.1], is below. This algorithm works only for Drinfeld modules where $\mathrm{CharPoly}(\phi_T) = \mathrm{MinPoly}_{\mathbb{F}_q}(\phi_T)$, which holds for generic choices of g and Δ; that is, for more than half of all elements of the parameter domain [22, theorem 3.6]. It further requires that the automorphism power projection algorithm of Kaltofen and Shoup can be extended elements of $L[X, \sigma]$. Narayanan stated the latter assumption as a fact, and although he was contacted regarding this statement, no resolution regarding whether it holds was reached.

We first state the following lemma due to Kaltofen and Saunders [17] :

Lemma 4. *Let A be an $n \times n$ matrix over a field F, $u : F^n \to F$ a linear map, and b a vector of length n whose entries come from a set $S \subset F$. Then:*

$$\mathrm{Prob}\big[\mathrm{MinPoly}(\{uA^ib\}_i) = \mathrm{MinPoly}(\{A^ib\}_i)\big] \geq 1 - \frac{\deg(\mathrm{MinPoly}(\{A^ib\}_i))}{2|S|}$$

and

$$\mathrm{Prob}\big[\mathrm{MinPoly}(\{A^ib\}_i) = \mathrm{MinPoly}(\{A^i\}_i)\big] \geq 1 - \frac{\deg(\mathrm{MinPoly}(\{A^i\}_i))}{2|S|}$$

Theorem 13. *Let $\phi = (\Delta, g)$ be a rank-2 Drinfeld module over L, $[L : \mathbb{F}_q] = n$, and suppose $\mathrm{CharPoly}(\phi_T) = \mathrm{MinPoly}_{\mathbb{F}_q}(\phi_T)$. There exists a Monte Carlo randomized algorithm for determining the characteristic polynomial $C_\phi = X^2 - aX + b$ in $O(n^{(\omega_2)/2+o(1)}\log^{1+o(1)} q + n^{1+o(1)}\log^{2+o(1)} q)$ $\mathbb{F}_q$ field operations.*

Proof. Choose an $\alpha \in L$ and an $\mathbb{F}_q$ linear map $\ell : L \to \mathbb{F}_q$ uniformly at random. By Lemma 4, with probability at least half and at least $1 - \frac{n}{2q}$, $\text{MinPoly}(\{\ell(\phi_T^i(\alpha)\}_i) = \text{MinPoly}_{\mathbb{F}_q}(\phi_T)$. The determination of $\text{MinPoly}(\{\ell(\phi_T^i(\alpha)\}_i)$ can be done using the Berlekamp-Massey algorithm in time $O(n^{1+o(1)} \log q)$ using the first $2n-1$ entries. Narayanan claims that computing the collection $\{\ell(\phi_T^i(\alpha)\}_{i=0}^{2n-2}$ is a restatement of the automorphism projection problem for elements of the endomorphism ring $L\{\sigma\}$, for which the algorithm of Kedlaya-Umans in [19] yields a runtime of $O(n^{(\omega_2)/2+o(1)} \log^{1+o(1)} q + n^{1+o(1)} \log^{2+o(1)} q)$ bit operations. □

5.5 New Algorithms for Computing the Characteristic Polynomial

5.5.1 A New Randomized Algorithm

Theorem 14. *Let $\phi = (\Delta, g)$ be a rank-2 Drinfeld module over L, $[L : \mathbb{F}_q] = n$. There exists a Monte Carlo randomized algorithm for determining the characteristic polynomial $C_\phi = X^2 - aX + b$ in $O(M(n)n \log q)$ $\mathbb{F}_q$ operations.*

Proof. Letting τ denote the Frobenius map σ^n, by theorem 8, the characteristic equation for τ tells us that for any $\alpha \in L$

$$\tau^2(\alpha) + \phi_b(\alpha) = \phi_a(\tau(\alpha))$$

Determining b using Gekeler's algorithm, we may compute the left-hand side efficiently by determining $\phi_T^i(\alpha)$ for $i \leq n$. Each individual application of ϕ_T requires three Frobenius operations on polynomials of degree n giving an individual runtime of $O(M(n) \log q)$ and a complexity of determining $\phi_b(\alpha)$ of $O(M(n)n \log q)$. Define $r := \alpha + \phi_b(\alpha)$ and $\ell : L \to \mathbb{F}_q$ a linear projection map, we can write down the following relations with

$$a = \sum_{i=0}^{\lfloor n/2 \rfloor} a_i T^i \in \mathbb{F}_q[T]$$

we get

$$r = \sum_{i=0}^{\lfloor n/2 \rfloor} a_i \phi_T^i(\alpha)$$

.

For $j \geq 0$ this implies

$$\ell(\phi_T^j(r)) = \sum_{i=0}^{\lfloor n/2 \rfloor} a_i \ell(\phi_T^{i+j}(\alpha)),$$

which gives us the following Hankel system H_κ for some parameter κ to be determined:

$$H_\kappa = \begin{bmatrix} \ell(\alpha) & \ell(\phi_T(\alpha)) & \dots & \ell(\phi_T^{\lfloor n/2 \rfloor}(\alpha)) \\ \vdots & \vdots & & \vdots \\ \ell(\phi_T^j(\alpha)) & \ell(\phi_T^{1+j}(\alpha)) & \dots & \ell(\phi_T^{\lfloor n/2 \rfloor + j}(\alpha)) \\ \vdots & \vdots & & \vdots \\ \ell(\phi_T^\kappa(\alpha)) & \ell(\phi_T^{1+\kappa}(\alpha)) & \dots & \ell(\phi_T^{\lfloor n/2 \rfloor + \kappa}(\alpha)) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_i \\ \vdots \\ a_{\lfloor n/2 \rfloor} \end{bmatrix} = \begin{bmatrix} \ell(r) \\ \ell(\phi_T(r)) \\ \vdots \\ \ell(\phi_T^j(r)) \\ \vdots \\ \ell(\phi_T^\kappa(r)) \end{bmatrix}$$

There are $O(\kappa + n)$ entries of the form $\ell(\phi_T^i(\alpha)), \ell(\phi_T^i(r))$ that need to be computed. Evaluating the projection map takes $O(n)$ time on each $\phi_T^i(\alpha)$. We have $\phi_T^{i+1}(\alpha) = (T + g\sigma + \Delta\sigma^2)\phi_T^i(\alpha)$ and evaluating the operator $T + g\sigma + \Delta\sigma^2$ on an element of L takes $O(M(n) \log q)$ time, giving a complexity of $O((n + \kappa)M(n) \log q)$

If $\kappa \in O(n)$, then the solution of the Hankel system can be solved in $O(n^2)$ operations using classical algorithms or $O(n \log^2(n))$ using algorithms due to Kaltofen [15]. It remains to show $\kappa \in O(n)$. This depends in part on a lemma of Kaltofen and Pan [16]

Lemma 5. *Let $\{a_i\}_{i=0}^\infty$ be a linear sequence over $\mathbb{F}_q$ and d the degree of its minimal polynomial. For any $m > 0$ let T_m be the matrix given by:*

$$\begin{bmatrix} a_0 & a_1 & \dots & a_{m-1} \\ a_1 & a_2 & \dots & a_m \\ \vdots & \vdots & \ddots & \vdots \\ a_{m-1} & a_m & \dots & a_{2m-2} \end{bmatrix}$$

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

Since L is a degree n extension of $\mathbb{F}_q$ there must be a degree n monic polynomial f over $\mathbb{F}_q$ such that $f(\phi_T) = 0$. Now let f be the minimal of ϕ_T with $\deg(f) = d$. For any positive integers i, j with $i < j < n$, $\sigma^i = \sigma^j$ implies σ^{j-i} is the identity on $\mathbb{F}_{q^n}$, which can't occur since $j - i < m$. Therefore by independence of characters, $\sigma, \sigma^2, \dots, \sigma^{n-1}$ satisfy no non-trivial $\mathbb{F}_q$-linear relation; that is, there are no constants $c_0, \dots, c_{n-1}$ with at least one $c_i \neq 0$ such that $c_0 + c_1\sigma + \dots + c_{n-1}\sigma^{n-1} = 0$. So if $0 = f(\phi_T) = c_0 + c_1\sigma + \dots + c_{2d}\sigma^{2d}$, where the lead term $c_{2d}\sigma^{2d}$ is given by $(\Delta\sigma^2)^d$, so $c_{2d} = \Delta^{(1-q^{2d})/(1-q)} \neq 0$, then $2d \geq n - 1$ and so $\frac{n-1}{2} \leq d = \deg \mathrm{MinPoly}_{\mathbb{F}_q}(\phi_T) \leq n$.

By lemma 4, with probability at least $(1 - \frac{n}{2q})^2$ we have that $\mathrm{MinPoly}_{\mathbb{F}_q}(\{\ell(\phi_T^i(\alpha)\}_i) = \mathrm{MinPoly}_{\mathbb{F}_q}(\phi_T)$, in which case by lemma 5 and the upper bound $\deg \mathrm{MinPoly}_{\mathbb{F}_q}(\phi_T) \leq n$ we can guarantee that an upper left submatrix of

$$\overline{H}_\kappa = \begin{bmatrix} \ell(\alpha) & \ell(\phi_T(\alpha)) & \dots & \ell(\phi_T^{\lfloor n/2 \rfloor}(\alpha)) & \ell(\phi_T^{\lfloor n/2 \rfloor+1}(\alpha)) & \dots & \ell(\phi_T^{2\lfloor n/2 \rfloor}(\alpha)) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ \ell(\phi_T^{j}(\alpha)) & \ell(\phi_T^{1+j}(\alpha)) & \dots & \ell(\phi_T^{\lfloor n/2 \rfloor+j}(\alpha)) & \ell(\phi_T^{\lfloor n/2 \rfloor+j+1}(\alpha)) & \dots & \ell(\phi_T^{2\lfloor n/2 \rfloor+j}(\alpha)) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ \ell(\phi_T^{\kappa}(\alpha)) & \ell(\phi_T^{1+\kappa}(\alpha)) & \dots & \ell(\phi_T^{\lfloor n/2 \rfloor+\kappa}(\alpha) & \ell(\phi_T^{\lfloor n/2 \rfloor+\kappa+1}(\alpha)) & \dots & \ell(\phi_T^{2\lfloor n/2 \rfloor+\kappa}(\alpha))) \end{bmatrix}$$

of size at least $\frac{n-1}{2}$ is invertible when $\kappa = \text{MinPoly}_{\mathbb{F}_q}(\{\ell(\phi_T^i(\alpha)\}_i)$ and $\frac{n-1}{2} \leq \kappa \leq n$. Therefore a solution to the system

$$\overline{H}_\kappa \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_i \\ \vdots \\ a_{\lfloor n/2 \rfloor} \\ a_0 \\ a_1 \\ \vdots \\ a_i \\ \vdots \\ a_{\lfloor n/2 \rfloor} \end{bmatrix} = \begin{bmatrix} \ell(r) + \ell(\phi_T^{\lfloor n/2 \rfloor}(r)) \\ \ell(\phi_T(r)) + \ell(\phi_T^{\lfloor n/2+1 \rfloor}(r)) \\ \vdots \\ \ell(\phi_T^{j}(r)) + +\ell(\phi_T^{\lfloor n/2 \rfloor+j}(r)) \\ \vdots \\ \ell(\phi_T^{\kappa}(r)) + \ell(\phi_T^{\lfloor n/2 \rfloor+\kappa}(r)) \end{bmatrix}$$

determines unique values for $a_0, \dots, a_{\lfloor n/2 \rfloor}$ with probability at least $(1 - \frac{n}{2q})^2$. We may determine the value of κ using the Berlekamp-Massey algorithm in $O(n^2)$ operations. One final observation is that when n is even and $\deg \text{MinPoly}_{\mathbb{F}_q}(\phi_T) = \frac{n}{2}$, then the invertible upper left matrix guaranteeing uniqueness has dimension $\frac{n}{2}$, which does not guarantee a unique solution for $a_{n/2}$. Using [12, proposition 2.14], this coefficient may be computed as

$$a_{n/2} = Tr_{\mathbb{F}_{q^2}/\mathbb{F}_q}(N_{L/\mathbb{F}_{q^2}}(\Delta)^{-1})$$

where $\mathbb{F}_{q^2}$ is the unique degree 2 extension of $\mathbb{F}_q$ contained in $\mathbb{F}_{q^n}$. Using $Tr_{\mathbb{F}_{q^2}/\mathbb{F}_q}(x) = x + x^q$ we can compute $a_{n/2}$ in $O(M(n)(\log n + \log q))$ $\mathbb{F}_q$ operations.

□

5.5.2 Schoof's Algorithm for Drinfeld modules

We present here an alternative approach inspired by Schoof's algorithm for elliptic curves. Supposing $\frac{n}{2} + 1 < q$, for some choice of elements $e_0, e_1, \dots, e_{n/2} \in \mathbb{F}_q$, we aim to compute

Algorithm 7 A New Randomized Algorithm for Finding the Frobenius Trace

procedure RANDOMIZEDFROBENIUSTRACE
  Input A rank-2 Drinfeld Module $(q, \mathfrak{f}, \mathfrak{p}, g, \Delta)$ and Frobenius Norm b
  Output The Frobenius Trace a
  Choose a map $\ell : L \to \mathbb{F}_q$ uniformly at random from $\mathbb{F}_q^n$
  Choose an element $\alpha \in L$ uniformly at random from L
  $r \leftarrow \alpha + \phi_b(\alpha)$
  $A[0] \leftarrow \alpha$
  $B[0] \leftarrow r$
  for $i = 1$ **to** $\lfloor \frac{n-1}{2} \rfloor$ **do**
    $B[i] = \phi_T(\phi_T^{i-1}(r))$
    $B[i-1] \leftarrow \ell(B[i-1])$
  for $i = 1$ **to** $n-1$ **do**
    $A[i] = \phi_T(\phi_T^{i-1}(\alpha))$
    $A[i-1] \leftarrow \ell(A[i-1])$
  for $i = 0$ **to** $\lfloor \frac{n-1}{2} \rfloor$ **do**
    for $j = 0$ **to** $\lfloor \frac{n}{2} \rfloor$ **do**
      $M[i,j] = A[i+j]$
  Solve Hankel system $Mx = B$ **return** x

$a(e_i)$. Invoking either the Universal Property of Quotients for rings, or the observation that $\phi_{a+r(T-e_i)} = \phi_a + \phi_r \phi_{T-e_i}$, this can be determined using $\phi_a \pmod{\phi_{T-e_i}} = \phi_{a(e_i)} = a(e_i)$ for each index i. Reducing the characteristic equation mod ϕ_{T-e_i} we obtain

$$a(e_i)\sigma^n = \sigma^{2n} + b(e_i) \pmod{\phi_T - e_i}. \tag{5.3}$$

Therefore it is sufficient to determine

$$\sigma^n \pmod{\phi_T - e_i}$$

for each e_i and solve for a satisfying equation 5.3.

Theorem 15. *Let $\phi = (\Delta, g)$ be a rank-2 Drinfeld module over L, $[L : \mathbb{F}_q] = n$. There exists a deterministic algorithm for determining the characteristic polynomial $C_\phi = X^2 - aX + b$ in*

- $O(n^{\omega_2/2+1} \log n + M(n) n \log q)$ $\mathbb{F}_q$ *operations*
- $O(n^{2+\delta} \log^{1+o(1)} q \log n)$ *bit operations for any $\delta > 0$.*

Proof. Let

$$\sigma^j = \nu_j + \mu_j \sigma \pmod{\phi_T - e_i}$$

for $\nu_j, \mu_j \in L$ and let $\gamma(T) = \gamma_T$. Then we obtain the recurrence relation:

$$\sigma^{j+1} = \nu_j^q \sigma + \mu_j^2 \sigma^2 = \nu_j^q \sigma + \mu_i^q \big(- \frac{\gamma_T - e_i}{\Delta} - \frac{g}{\Delta} \sigma \big) \pmod{\phi_T - e_i}.$$

That is, $\nu_{j+1} = -\frac{\gamma_T - e_i}{\Delta} \mu_j^q$ and $\mu_{j+1} = \nu_j^q - \frac{g}{\Delta} \mu_j^q$. Letting $\alpha = -\frac{\gamma_T - e_i}{\Delta}$, $\beta = -\frac{g}{\Delta}$, and

$$M^{(q^i)} = \begin{bmatrix} 0 & \alpha^{q^i} \\ 1 & \beta^{q^i} \end{bmatrix}$$

$$\begin{bmatrix} \nu_n \\ \mu_n \end{bmatrix} = M M^{(q)} \dots M^{(q^{n-1})} \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Therefore, efficiently computing σ^n relies on computing the products $P_j = M M^{(q)} \dots M^{(q^j)}$. This can be done effectively using the following observation:

$$P_{2j+1} = P_j P_j^{(q^{j+1})}.$$

We then compute σ^{2n} as

$$\begin{aligned} \sigma^{2n} &= (\nu_n + \mu_n \sigma)(\nu_n + \mu_n \sigma) && \pmod{\phi_T - e_i} \\ &= \nu_n^2 + \nu_n \mu_n \sigma + \mu_n \nu_n^q \sigma + \mu_n^{q+1} (\alpha + \beta \sigma) && \pmod{\phi_T - e_i} \\ &= \nu_n^2 + \mu_n^{q+1} \alpha + (\nu_n \mu_n + \mu_n \nu_n^q + \mu_n^{q+1} \beta) \sigma && \pmod{\phi_T - e_i} \end{aligned}$$

Substituting back into 5.3 results in the relation

$$a(e_i)(\nu_n + \mu_n \sigma) = \nu_n^2 + \mu_n^{q+1} \alpha + b(e_i) + (\nu_n \mu_n + \mu_n \nu_n^q + \mu_n^{q+1} \beta) \sigma \pmod{\phi_T - e_i}. \quad (5.4)$$

Therefore when $\mu_n \neq 0$ we conclude that

$$a(e_i) = \nu_n + \nu_n^q + \mu_n^q \beta$$

which allows the determination of $a(e_i)$ without computing the Frobenius norm beforehand. Otherwise, 5.4 gives

$$a(e_i) = \nu_n + b(e_i) \quad (5.5)$$

when $\mu_n = 0$.

The algorithm may be summarized as follows:

1. Compute $\sigma^n \pmod{\phi_T - e_i}$ by computing P_n.

2. Solve for $a(e_i) \in \mathbb{F}_q$ such that $a(e_i)\sigma^n = \sigma^{2n} + b(e_i) \pmod{\phi_T - e}$.

3. Interpolate the values $a(e_i)$.

The only potential degeneracy occurs when $\sigma^n = 0 \pmod{\phi_T - e_i}$, which can be excluded when $\gamma_T \neq e_i$. Suppose $\sigma^n = \sum_{i=0}^{n-2} a_i \sigma^i (\phi_T - e_i)$. Then clearly $a_0 = 0$ if $\gamma_T - e \neq 0$. Moreover, if $a_i = 0$ for all $i < d < n-2$, then $0 = a_d(\gamma_T - e_i)^d + a_{d-1}g^{d-1} + a_{d-2}\Delta^{d-2} = a_d(\gamma_T - e_i)^d$, therefore $a_d = 0$. So $\sigma^n = a_{n-2}\sigma^{n-2}\sigma^i(\phi_T - e_i)$ which is a contradiction.

We can compute polynomial representations for all σ^i beforehand using modular composition for at most $O(\log n)$ choices of index i. Next, for each e_i, we perform $O(\log n)$ 2×2 matrix multiplications and modular compositions to evaluate the Frobenius map up to order $q^{n/2}$. These two steps together contribute $O(n^{\omega_2/2}\log n + M(n)\log q)$ $\mathbb{F}_q$ operations using Brent-Kung composition, or $O(n^{1+\delta}\log^{1+o(1)} q \log n)$ bit operations, for any $\delta > 0$, using Kedlaya-Umans. Repeating for $\frac{n}{2}+1$ choices of e_i raises the cost in either case by a factor of $O(n)$. The interpolation step takes at most $O(n^2)$ operations, so the total cost remains either $O(n^{\omega_2/2+1}\log n + M(n)n\log q)$ $\mathbb{F}_q$ operations or $O(n^{2+\delta}\log^{1+o(1)} q \log n)$ bit operations

□

We expect the algorithm may be extended to the case $\frac{n}{2} + 1 > q$ by computing $\sigma^n \bmod \phi_g$ for irreducible polynomials g, though an exposition of this approach will not be given.

Example 7. *Let $q = 5$, $n = 4$. Then $L = \mathbb{F}_5[T]/(T^4 + 4T^2 + 4T + 2)$. Take ϕ to be a rank-2 Drinfeld module over L with $g = 1, \Delta = 1$. Then for $e_0 = 0$ we have $\alpha = 4t$, $\beta = 4$. Letting*

$$M = \begin{bmatrix} 0 & 4T \\ 1 & 4 \end{bmatrix}.$$

We get:

$$MM^5M^{25}M^{125} = \begin{bmatrix} T^3 + T^2 + 3 & T^3 + 2T + 3 \\ 2T^3 + 2T^2 + 3T + 4 & 4T^3 + 4T^2 + 4 \end{bmatrix}$$

So $\nu_4 = T^3 + T^2 + 3$ and $\mu_4 = 2T^3 + 2T^2 + 3T + 4$ and

$$a(0) = \nu_4 + \nu_4^5 + 4\mu_4^5 = 2$$

For $e_1 = 1$ we have

Algorithm 8 Schoof's Algorithm for Drinfeld modules

procedure FROBENIUSTRACE
  Input A rank-2 Drinfeld Module $(q, \mathfrak{f}, \mathfrak{p}, g, \Delta)$ and Frobenius Norm b
  Output The Frobenius Trace a
  $E \leftarrow \{e_0, \ldots, e_{n/2}\} \subset \mathbb{F}_q$
  $\sigma \leftarrow x^q \bmod \mathfrak{f}$
  while $i < n$ **do**
    if $2i + 1 < n$ **then**
      $\sigma^{i+1} \leftarrow \sigma^i \circ \sigma$
      $\sigma^{2i+1} \leftarrow \sigma^i \circ \sigma^i \circ \sigma$
      $i \leftarrow 2i + 1$
    else
      $\sigma^{i+1} \leftarrow \sigma^i \circ \sigma$
      $i \leftarrow i + 1$
  for $e_i \in E$ **do**
    $\alpha \leftarrow -(\gamma_T - e_i)/\Delta$
    $\beta \leftarrow -g/\Delta$
    $P_0 \leftarrow \begin{bmatrix} 0 & \alpha \\ 1 & \beta \end{bmatrix}$
    $i \leftarrow 0$
    while $i < n - 1$ **do**
      if $2i + 1 < n$ **then**
        $P_i^{\sigma^{i+1}} \leftarrow \begin{bmatrix} P_i[0,0](\sigma^{i+1}) & P_i[0,1](\sigma^{i+1}) \\ P_i[1,0](\sigma^{i+1}) & P_i[1,1](\sigma^{i+1}) \end{bmatrix}$
        $P_{2i+1} \leftarrow P_i P_i^{\sigma^{i+1}}$
        $i \leftarrow 2i + 1$
      else
        $M_i \leftarrow \begin{bmatrix} 0 & \alpha(\sigma^i) \\ 1 & \beta(\sigma^i) \end{bmatrix}$
        $P_{i+1} \leftarrow P_i M_i$
        $i \leftarrow i + 1$
    $\nu_n \leftarrow P_{n-1}[0,0]$
    $\mu_n \leftarrow P_{n-1}[1,0]$
    if $\mu_n \neq 0$ **then**
      $a(e_i) \leftarrow \nu_n + \nu_n^q + \mu_n^q \beta$
    else
      $a(e_i) \leftarrow \nu_n + b(e_i)$
  return Interpolate$(\{(e_0, a(e_0)), (e_1, a(e_1)), \ldots, (e_{n/2}, a(e_{n/2}))\})$

$$M = \begin{bmatrix} 0 & 4t+1 \\ 1 & 4 \end{bmatrix}$$
$$MM^5M^{25}M^{125} = \begin{bmatrix} 4T & 4T^2 \\ 2T^3+2T^2+3T+2 & T+3 \end{bmatrix}$$

and we get $a(1) = 3$. *Repeating this calculation once more yields* $a(2) = 3$ *and we can interpolate to get*

$$a = (T-1)(T-2) + 2T(T-2) + 4T(T-1) = 2T^2 + 4T + 2.$$

Chapter 6

Computational Results

We attempted to verify the runtime of the two new algorithms presented here with an implementation using SageMath [27]. The rank-2 Drinfeld module with parameters $g = 1$, $\Delta = 1$ was used for all computations. The runtimes shown in tables 6.1 and 6.2 are for fields of characteristic $p = 1299827$ and $p = 179426549$ for varying values of k such that the base field has order $q = p^k$, and $n = [L : \mathbb{F}_q]$ was fixed at $n = 6$. The values given are the averages over 10 trials. The results appear to confirm the logarithmic dependence on q for both new algorithms.

	$p = 1299827$	$p = 179426549$
$k = 1$	0.0791	0.0951
$k = 3$	1.1438	1.418019056
$k = 6$	5.3689	11.35429311

Table 6.1: Runtime in seconds of our new randomized algorithm when $q = p^k$ and $n = 6$

	$p = 1299827$	$p = 179426549$
$k = 1$	0.0791	0.0951
$k = 3$	1.1438	1.418019056
$k = 6$	5.3689	11.35429311

Table 6.2: Runtime in seconds of our new deterministic algorithm when $q = p^k$ and $n = 6$

We then attempted to verify the order of dependence on n by producing a log-log plot on n versus runtime. Logarithms are taken base 2, and results are averaged over 10 trials. The linear regressions for our randomized algorithm shown in figures 6 and 6 have slopes below 3, which is consistent with the sub-cubic runtime given by our analysis.

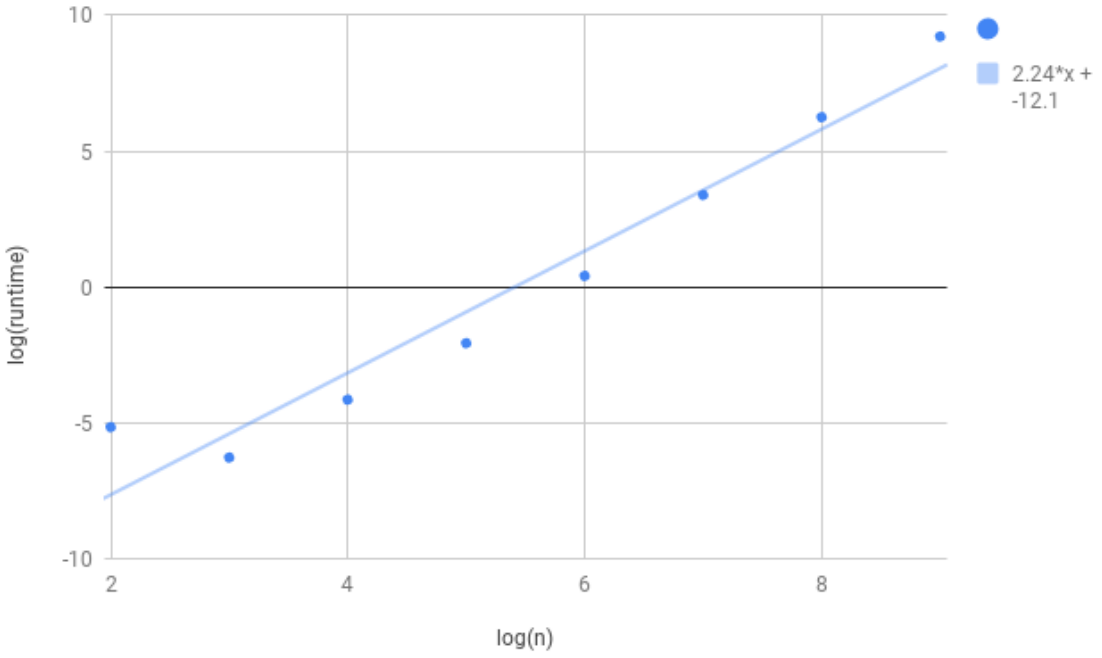

Figure 6.1: Log-log plot of n versus runtime with $p = 2$, $k = 1$ for our new randomized algorithm

In contrast, our linear regression for the deterministic algorithm exceeds 3. This is almost certainly due to difficulties in finding library implementations for fast modular composition. The only reference implementations we could find worked only in the case where the base field was $\mathbb{F}_2$, which due to the requirement $q > \frac{n}{2} + 1$ was insufficient for testing our deterministic algorithm. Efforts to produce our own general implementation of fast modular composition were unsuccessful in the time available.

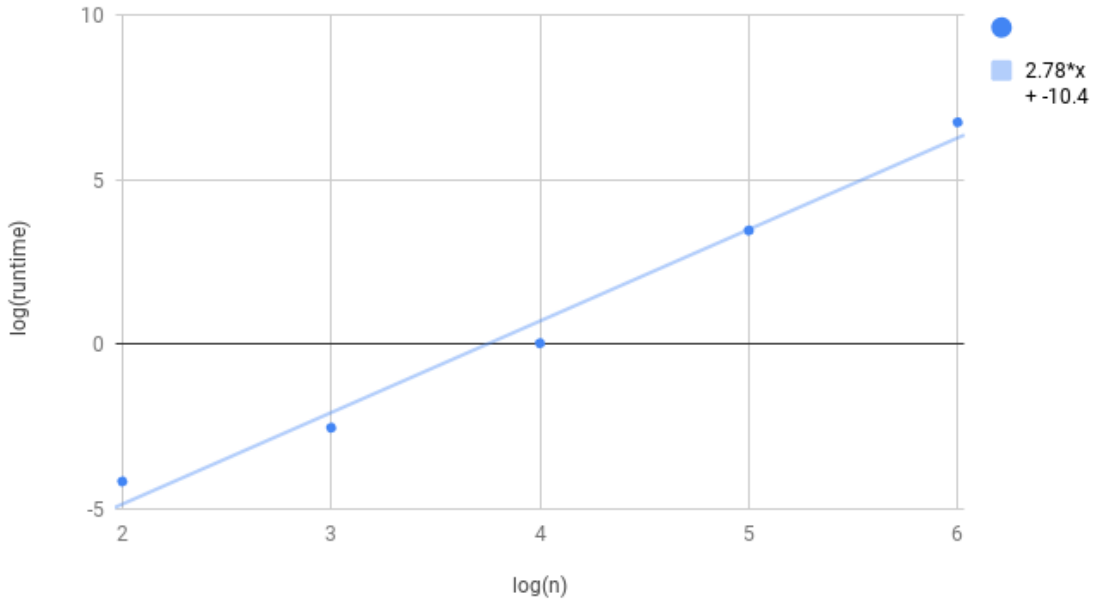

Figure 6.2: Log-log plot of n versus runtime with $p = 31$, $k = 2$ for our new randomized algorithm

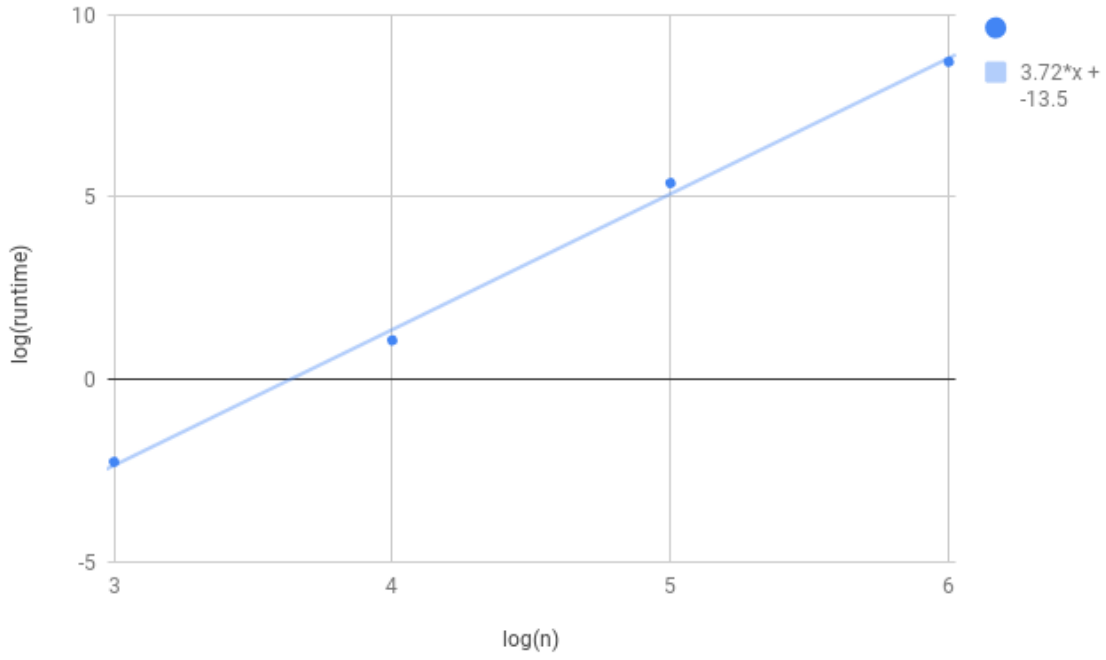

Figure 6.3: Log-log plot of n versus runtime with $p = 31$, $k = 2$ for our new deterministic algorithm

Bibliography

[1] Alin Bostan, Bruno Salvy, Francois Morain, and Eric Schost. Fast algorithms for computing isogenies between elliptic curves. Research report, 2006.

[2] R. P. Brent and H. T. Kung. Fast algorithms for manipulating formal power series. *J. ACM*, 25(4):581–595, October 1978.

[3] Anne Canteaut. *Berlekamp-Massey algorithm*, pages 29–30. Springer US, Boston, MA, 2005.

[4] David G. Cantor and Erich Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28:693–701, 1991.

[5] Xavier Caruso and Jérémy Le Borgne. Fast multiplication for skew polynomials. *CoRR*, abs/1702.01665, 2017.

[6] J. Doliskani, A. K. Narayanan, and É. Schost. Drinfeld Modules with Complex Multiplication, Hasse Invariants and Factoring Polynomials over Finite Fields. *ArXiv e-prints*, December 2017.

[7] V. G. Drinfel'd. Elliptic modules. *Matematicheskii Sbornik*, 94(23):561593, 1974.

[8] W. Fulton. *Algebraic curves: an introduction to algebraic geometry*. Advanced book classics. Addison-Wesley Pub. Co., Advanced Book Program, 1989.

[9] François Le Gall and Florent Urrutia. Improved rectangular matrix multiplication using powers of the coppersmith-winograd tensor. *CoRR*, abs/1708.05622, 2017.

[10] Joachim Von Zur Gathen and Jurgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[11] Ernst-Ulrich Gekeler. On finite drinfeld modules. *Journal of Algebra*, 141(1):187 – 203, 1991.

[12] Ernst-Ulrich Gekeler. Frobenius distributions of drinfeld modules over finite fields. 360:1695–1721, 04 2008.

[13] M. Giesbrecht. Factoring in skew-polynomial rings over finite fields. *Journal of Symbolic Computation*, 26(4):463 – 486, 1998.

[14] Xiaohan Huang and Victor Y. Pan. Fast rectangular matrix multiplication and applications. *Journal of Complexity*, 14(2):257 – 299, 1998.

[15] Erich Kaltofen. Asymptotically fast solution of toeplitz-like singular linear systems. pages 297–304, 01 1994.

[16] Erich Kaltofen and Victor 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, pages 180–191, New York, NY, USA, 1991. ACM.

[17] Erich Kaltofen and B. David Saunders. On wiedemann's method of solving sparse linear systems. In *Proceedings of the 9th International Symposium, on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, AAECC-9, pages 29–38, London, UK, UK, 1991. Springer-Verlag.

[18] Erich Kaltofen and Victor Shoup. Subquadratic-time factoring of polynomials over finite fields. In *Proceedings of the Twenty-seventh Annual ACM Symposium on Theory of Computing*, STOC '95, pages 398–406, New York, NY, USA, 1995. ACM.

[19] Kiran S. Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, December 2011.

[20] F. Le Gall. Powers of Tensors and Fast Matrix Multiplication. *ArXiv e-prints*, January 2014.

[21] J. Massey. Shift-register synthesis and bch decoding. *IEEE Transactions on Information Theory*, 15(1):122–127, January 1969.

[22] Anand Kumar Narayanan. Polynomial factorization over finite fields by computing euler-poincare characteristics of drinfeld modules. *CoRR*, abs/1504.07697, 2015.

[23] M. Pohst and H. Zassenhaus, editors. *Algorithmic Algebraic Number Theory*. Cambridge University Press, New York, NY, USA, 1989.

[24] Sven Puchinger and Antonia Wachter-Zeh. Fast operations on linearized polynomials and their applications in coding theory. *Journal of Symbolic Computation*, 2017.

[25] Thomas Scanlon. Public key cryptosystems based on drinfeld modules are insecure. *Journal of Cryptology*, 14(4):225–230, Sep 2001.

[26] René Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Mathematics of Computation*, 44(170):483–494, 1985.

[27] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version x.y.z)*, YYYY. `http://www.sagemath.org`.

[28] Joachim von zur Gathen and Mark Giesbrecht. Constructing normal bases in finite fields. *J. Symb. Comput.*, 10:547–570, 1990.

[29] Joachim von zur Gathen and Victor Shoup. Computing frobenius maps and factoring polynomials. *computational complexity*, 2(3):187–224, Sep 1992.

[30] D H Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theor.*, 32(1):54–62, January 1986.