Computing Zeta Functions of Cyclic Covers of $\mathbb{P}^1$ in Large Characteristic

Vishal Arul, Alex J. Best, Edgar Costa, Richard Magner, Nicholas Triantafillou

MIT, Boston U., Dartmouth/MIT, Boston U., MIT

17th July 2018
Thirteenth Algorithmic Number Theory Symposium
University of Wisconsin, Madison

Notation/Goal

- $\mathbb{F}_q$ is the finite field with $q = p^n$ elements.
- $\overline{F} \in \mathbb{F}_q[x]$ is a square-free polynomial of degree d.
- $\mathcal{C}$ is the cyclic cover of $\mathbb{P}^1$ of degree r with affine model $y^r = \overline{F}(x)$.

$$g = \frac{rd - r - d - \gcd(r, d)}{2} + 1.$$

Goal:

Compute

$$Z(\mathcal{C}, t) := \exp\left(\sum_{i=1}^{\infty} \#\mathcal{C}(\mathbb{F}_{q^i})\frac{t^i}{i}\right) = \frac{\det\left(1 - t \cdot \mathrm{Frob}_q \,|H^1(\mathcal{C})\right)}{(1-t)(1-qt)},$$

as quickly as possible (in theory and practice!)

Why Compute Zeta Functions of Cyclic Covers?

Zeta Functions: Accumulate knowledge about arithmetic curves.

- Sato-Tate.
- Lang-Trotter.
- Torsion subgroups of Jacobians.
- Galois representations.
- Much more!

Cyclic Covers:

- Extra endomorphisms.
- Understand what features of hyperelliptic curves are used.
- Test our computational reach.

Main Result

Theorem

Suppose $p > d^2 r^2 n/2 + \log_p(dr) + 2$.
Let $\overline{F} \in \mathbb{F}_{p^n}[x]$ *be a square-free polynomial of degree* d.
Let $\mathcal{C}$ *be the smooth projective curve with affine model*

$$\mathcal{C} : y^r = \overline{F}(x).$$

The zeta function of $\mathcal{C}$ *can be computed in time*

$$O\left(p^{1/2} \cdot \textit{Polynomial in } n, r, d, \log p\right).$$

We implemented our method in Sage. It performs well in practice.

Our examples were computed on one core of a desktop machine with an `Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz`.

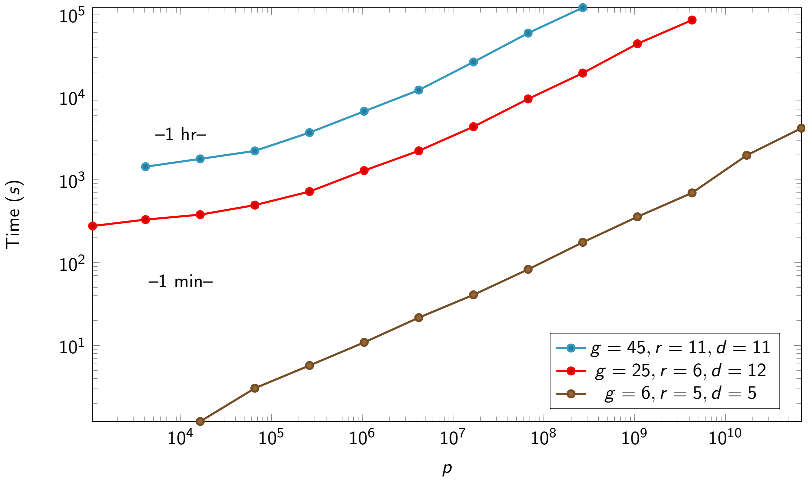


Figure: Timings on a log-log plot. Time is roughly proportional to $p^{1/2}$.

History - Computing Zeta Fuctions

- p-adic cohomology approach - variants of Kedlaya's algorithm
  - $p^{1/2+\varepsilon}$ or average polynomial in $\log p$ over many primes, polynomial in genus g.
  - Hyperelliptic/superelliptic versions are efficient in practice.
- Other approaches:
  - ℓ-adic approach - variants of Schoof's method.
  - Deformation theory
  - Trace formulas
- The dream:
  - Algorithm polynomial in $\log p$ and g simultaneously.

History of Kedlaya-style algorithms

Theorem

(Kedlaya 2001)
Let $\overline{F} \in \mathbb{F}_q[x]$ be a monic square-free polynomial of degree d.
Let $\mathcal{C}$ be the smooth projective curve with affine model

$$\mathcal{C} : y^r = \overline{F}(x).$$

When $r = 2$ and d is odd, the zeta function of $\mathcal{C}$ can be computed in time

$$O\left(p \cdot \text{Polynomial in } n, r, d, \log p\right).$$

History of Kedlaya-style algorithms

Theorem

(Harvey 2007)
Let $\overline{F} \in \mathbb{F}_q[x]$ be a monic square-free polynomial of degree d.
Let $\mathcal{C}$ be the smooth projective curve with affine model

$$\mathcal{C} : y^r = \overline{F}(x).$$

When $r = 2$ and d is odd, the zeta function of $\mathcal{C}$ can be computed in time

$$O\left(p^{1/2} \cdot \textit{Polynomial in } n, r, d, \log p\right).$$

History of Kedlaya-style algorithms

Theorem

(Minzlaff 2010)
Let $\overline{F} \in \mathbb{F}_q[x]$ be a monic square-free polynomial of degree d.
Let $\mathcal{C}$ be the smooth projective curve with affine model

$$\mathcal{C} : y^r = \overline{F}(x).$$

When $\gcd(r, d) = 1$, the zeta function of $\mathcal{C}$ can be computed in time

$$O\left(p^{1/2} \cdot \textit{Polynomial in } n, r, d, \log p\right).$$

History of Kedlaya-style algorithms

Theorem

(Gonçalves 2015)
Let $\overline{F} \in \mathbb{F}_q[x]$ *be a monic square-free polynomial of degree* d*.*
Let $\mathcal{C}$ *be the smooth projective curve with affine model*

$$\mathcal{C} : y^r = \overline{F}(x).$$

For any r, d*, the zeta function of* $\mathcal{C}$ *can be computed in time*

$$O\left(p \cdot \textit{Polynomial in } n, r, d, \log p\right).$$

History of Kedlaya-style algorithms

Theorem

(ABCMT 2018)
Let $\overline{F} \in \mathbb{F}_q[x]$ *be any square-free polynomial of degree* d*.*
Let $\mathcal{C}$ *be the smooth projective curve with affine model*

$$\mathcal{C} : y^r = \overline{F}(x).$$

For any r, d*, the zeta function of* $\mathcal{C}$ *can be computed in time*

$$O\left(p^{1/2} \cdot \text{Polynomial in } n, r, d, \log p\right).$$

What is $Z(\mathcal{C}, t)$?

The numerator of $Z(\mathcal{C}, t)$ is

$$\det\left(1 - t \cdot \mathrm{Frob}_q \,|H^1(\mathcal{C})\right).$$

We use Monsky-Washnitzer cohomology of the punctured curve

$$\widetilde{\mathcal{C}} := \{y^r = \overline{F}(x)\} \setminus (\{y = 0\} \cup \{\text{pts at } \infty\})$$

to compute this numerator.

$$H^1(\widetilde{\mathcal{C}}) = \mathbb{Q}_q^\dagger[[x, y^{-1}]]dx/(\text{Relations}).$$

The relations come from manipulating the equation $y^r - F(x) = 0$.

Monomial Basis:

$$B_\epsilon := \left\{\frac{x^i}{y^j}dx : i \in \{0, \ldots, d-2\}, j \in \{\epsilon r + 1, \ldots, (\epsilon + 1)r - 1\}\right\}$$

Overview of Kedlaya-style algorithms

1. Compute action of Frobenius on Monsky-Washnitzer H^1. Get a matrix with respect to B_0.
    1. Expand $\mathrm{Frob}(x^i dx/y^j)$ as a power series.
    2. Truncate the power series.
    3. 'Reduce' to a linear combination of basis elements.
2. Find the characteristic polynomial.

[From previous slide] Monomial Basis:

$$B_0 := \left\{ \frac{x^i}{y^j} dx : i \in \{0, \ldots, d-2\}, j \in \{1, \ldots, r-1\} \right\}.$$

Overview of Kedlaya-style algorithms

1. Compute action of Frobenius on Monsky-Washnitzer H^1. Get a matrix with respect to B_0.
    1. Expand $\mathrm{Frob}(x^i dx/y^j)$ as a power series.
    2. Truncate the power series.
    3. ‘Reduce’ to a linear combination of basis elements.
2. Find the characteristic polynomial.
3. Divide out a factor corresponding to the punctures at infinity.
    - Degree $= \gcd(r, d) - 1$.
    - Depends on $\gcd(r, d)$ and the leading coefficient of F.
    - Easy to compute.

[From previous slide] Monomial Basis:

$$B_0 := \left\{ \frac{x^i}{y^j} dx : i \in \{0, \ldots, d-2\}, j \in \{1, \ldots, r-1\} \right\}.$$

Overview of Kedlaya-style algorithms

1. Compute action of Frobenius on Monsky-Washnitzer H^1. Get a matrix with respect to B_1.
   1. Expand $\mathrm{Frob}(x^i dx/y^j)$ as a power series.
   2. Truncate the power series.
   3. 'Reduce' to a linear combination of basis elements.
2. Find the characteristic polynomial.
3. Divide out a factor corresponding to the punctures at infinity.
   - Degree $= \gcd(r, d) - 1$.
   - Depends on $\gcd(r, d)$ and the leading coefficient of F.
   - Easy to compute.

[From previous slide] Monomial Basis:

$$B_1 := \left\{ \frac{x^i}{y^j} dx : i \in \{0, \ldots, d-2\}, j \in \{r+1, \ldots, 2r-1\} \right\}.$$

Expanding Frob $\left(\frac{x^i}{y^j}dx\right)$

$$\text{Frob}\left(\frac{x^i}{y^j}dx\right) = \frac{px^{pi+p-1}}{y^{jp}} \sum_{k=0}^{\infty} \binom{-j/r}{k} \left(\frac{F(x^p)}{y^{pr}} - 1\right)^k dx.$$

Key Features:

- For p-adic precision N, only need $N+1$ terms.

Truncating the power series

$$\text{Frob}\left(\frac{x^i}{y^j}dx\right) = \frac{px^{pi+p-1}}{y^{jp}} \sum_{k=0}^{N} \binom{-j/r}{k} \left(\frac{F(x^p)}{y^{pr}} - 1\right)^k dx.$$

Key Features:

- For p-adic precision N, only need $N+1$ terms.
- Sparse - Only $\approx \frac{1}{2}N^2 d$ monomials $x^s dx/y^t$ have non-zero coefficients.
- Exponents are still big!

‘Reducing differentials’

Problem:	
Have:	Differentials $\frac{x^s}{y^t}dx$ for s, t large.
Want:	Cohomologous differentials $\sum_{i,j} a_{i,j}\frac{x^i}{y^j}dx$ for i, j small.
Solution:	Use relations in cohomology to ‘reduce’ $\frac{x^s}{y^t}dx$ to linear combinations of differentials with smaller exponents.

We use two types of relations:

- Horizontal Reduction – reduces x-degree.
- Vertical Reduction – reduces y-degree.

Reductions

There is a relation:

$$\frac{x^s \cdot x^{d-1}}{y^t} dx \sim \frac{x^{s-1} \cdot (\text{Degree } d-1 \text{ polynomial in } x)}{(d(t-r)-rs)y^t} dx.$$

Note that the x-degree goes down and the y-degree is unchanged.

Reductions

There is a relation:

$$\frac{x^s \cdot x^{d-1}}{y^t} dx \sim \frac{x^{s-1} \cdot (\text{Degree } d-1 \text{ polynomial in } x)}{(d(t-r)-rs)y^t} dx.$$

Note that the x-degree goes down and the y-degree is unchanged.

The denominator $(d(t-r)-rs)$ could be zero if $(r,d) \neq 1$.

Reductions

There is a relation:

$$\frac{x^s \cdot x^{d-1}}{y^t} dx \sim \frac{x^{s-1} \cdot (\text{Degree } d-1 \text{ polynomial in } x)}{(d(t-r)-rs)y^t} dx.$$

Set

$$W_{s,t} := \operatorname{span}_{\mathbb{Q}_p} \left(\frac{x^{s+i}}{y^t} dx : 0 \le i \le d-1 \right).$$

Horizontal Reduction: When $d(t-r)-rs \neq 0$, the relation above induces a linear map $W_{s,t} \to W_{s-1,t}$ preserving cohomology class.

Reductions

There is a relation:

$$\frac{x^s \cdot x^{d-1}}{y^t}dx \sim \frac{x^{s-1} \cdot (\text{Degree } d-1 \text{ polynomial in } x)}{(d(t-r)-rs)y^t}dx.$$

Set

$$W_{s,t} := \text{span}_{\mathbb{Q}_p}\left(\frac{x^{s+i}}{y^t}dx : 0 \leq i \leq d-1\right).$$

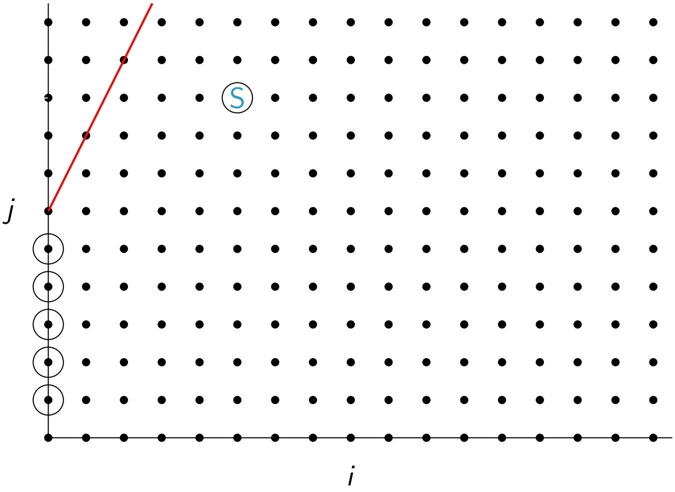
Horizontal Reduction: When $d(t-r)-rs \neq 0$, the relation above induces a linear map $W_{s,t} \to W_{s-1,t}$ preserving cohomology class.

Vertical Reduction: When $t \neq r$, other relations induce linear maps $W_{0,t} \to W_{0,t-r}$ preserving cohomology class.

‘Picture of Reduction’ for $r = 6, d = 3$.

The point (i, j) represents $W_{i,j}$.

Horizontal reduction fails on the red line.

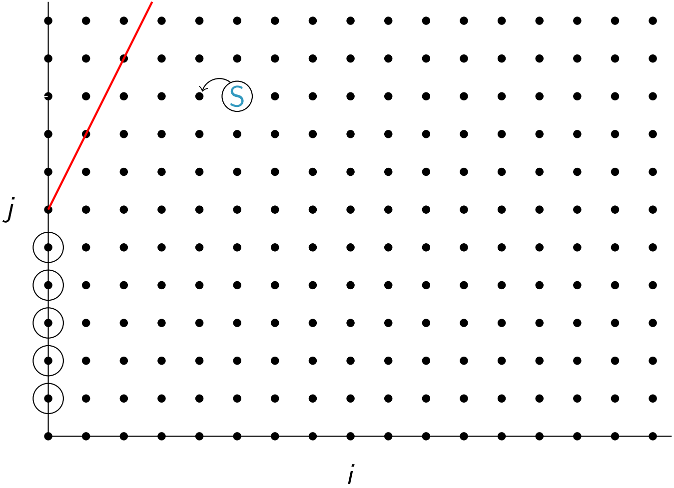


'Picture of Reduction' for $r = 6, d = 3$.

The point (i, j) represents $W_{i,j}$.

Horizontal reduction fails on the red line.

First reduce horizontally.

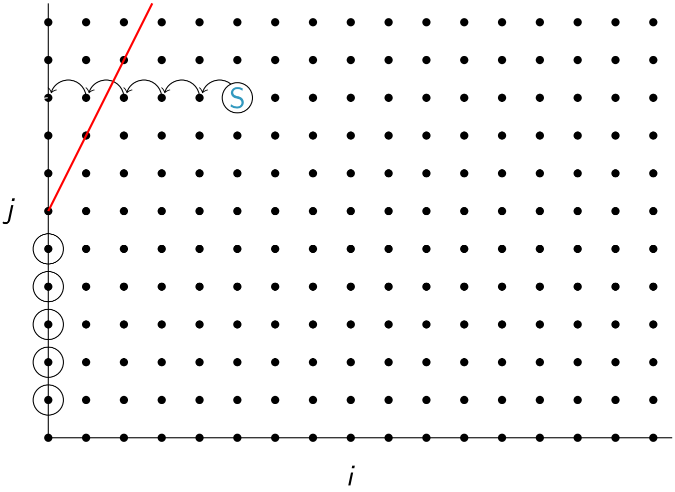


‘Picture of Reduction’ for $r = 6, d = 3$.

The point (i, j) represents $W_{i,j}$.

Horizontal reduction fails on the red line.

First reduce horizontally.

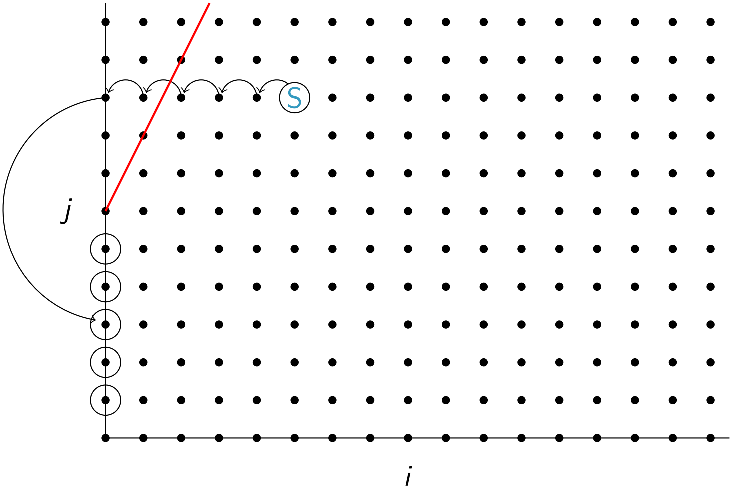


'Picture of Reduction' for $r = 6, d = 3$.

The point (i, j) represents $W_{i,j}$.

Horizontal reduction fails on the red line.

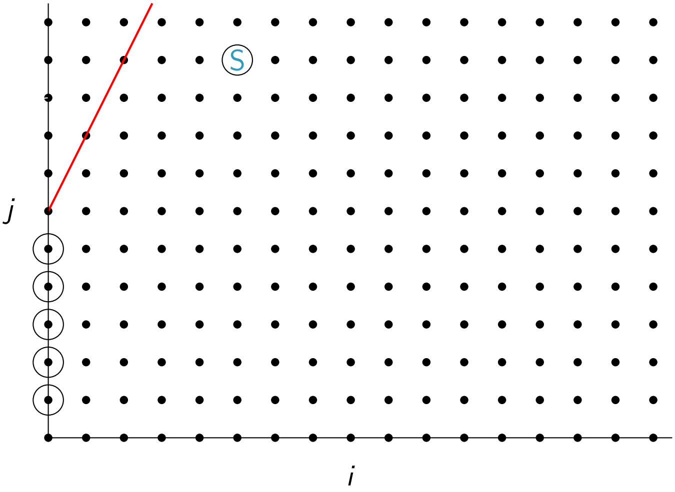
First reduce horizontally. Then reduce vertically.



'Picture of Reduction' for $r = 6, d = 3$.

The point (i, j) represents $W_{i,j}$.

Horizontal reduction fails on the red line.

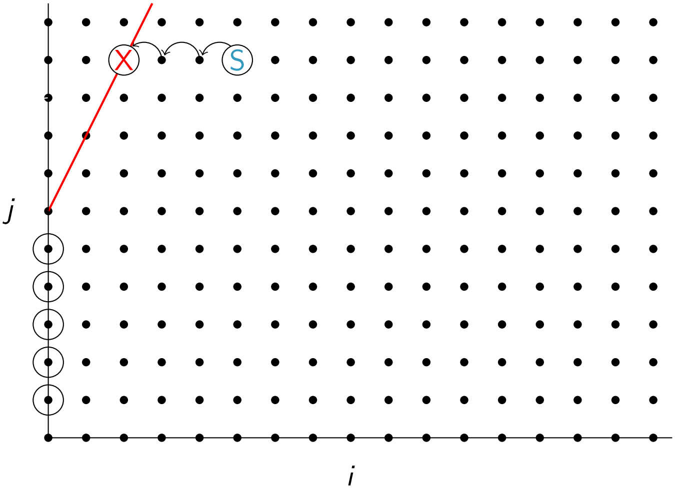


‘Picture of Reduction’ for $r = 6, d = 3$.

The point (i, j) represents $W_{i,j}$.

Horizontal reduction fails on the red line.

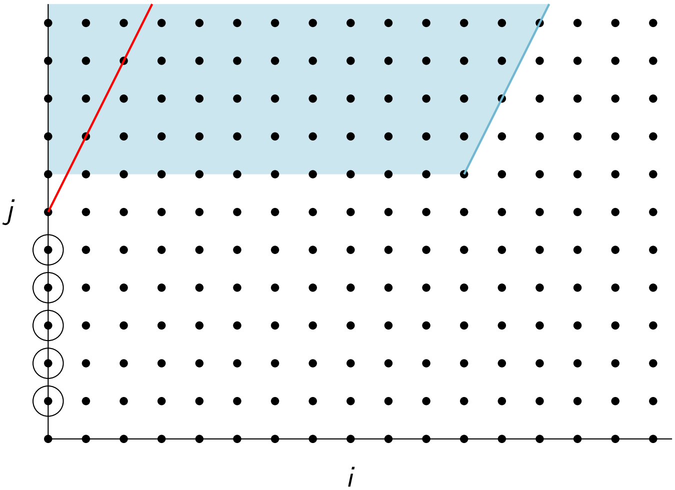
We can’t reduce horizontally!



‘Picture of Reduction’ for $r = 6, d = 3, p = 7$.

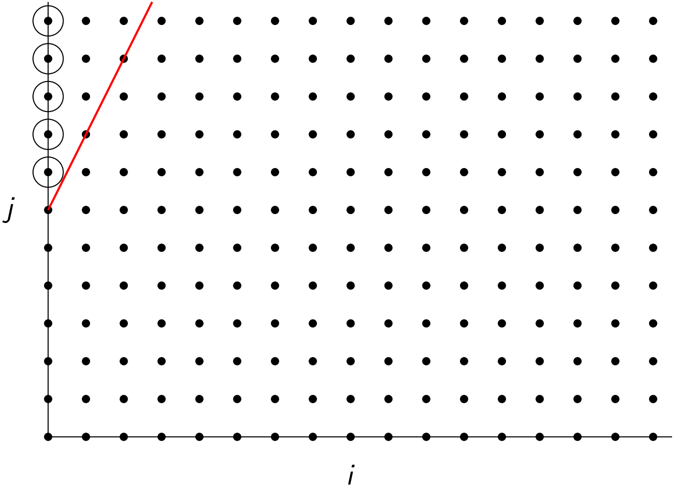
The shaded region shows where the terms to reduce live if we use the naive basis $B_0 := \left\{ \frac{x^i}{y^j} dx : 0 \leq i \leq d-2, 1 \leq j \leq r-1 \right\}$.

This is a real problem when $(r, d) \neq 1$.



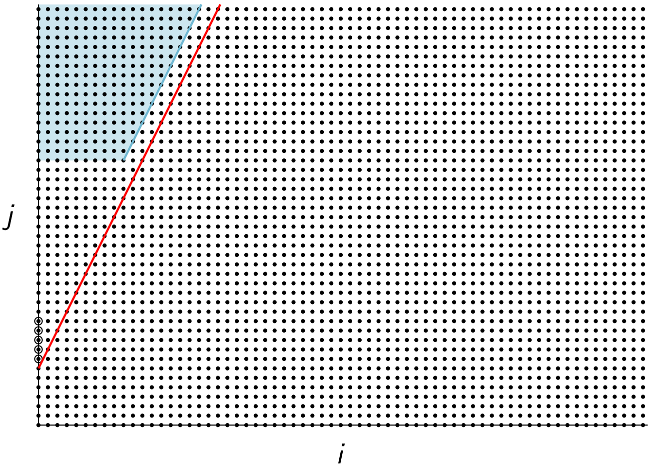
‘Picture of Reduction’ for $r = 6, d = 3, p = 7$.

If we use basis $B_1 := \left\{ \frac{x^i}{y^j} dx : 0 \le i \le d-2, r+1 \le j \le 2r-1 \right\}$, all terms lie to the left of the red line, so we can always reduce.



‘Picture of Reduction’ for $r = 6, d = 3, p = 7$.

If we use basis $B_1 := \left\{ \frac{x^i}{y^j} dx : 0 \leq i \leq d-2, r+1 \leq j \leq 2r-1 \right\}$,
all terms lie to the left of the red line, so we can always reduce.



$O(p^{1/2+\varepsilon})$ speed-up.

Reducing naively, we have an $O(p)$ algorithm.

Reduction matrices are in linear progressions. Multiplying them with Bostan-Gaudry-Schost, as in Harvey or Minzlaff, gives an $O(p^{1/2+\epsilon})$ algorithm.

Technical Disclaimer:
Many crucial details have been swept under the rug.
E.g. Applying Bostan-Gaudry-Schost carefully allows us to do all computations with only one extra digit of p-adic precision.

Restatement of Main Result

Theorem

Suppose $p > d^2 r^2 n/2 + \log_p(dr) + 2$.
Let $\overline{F} \in \mathbb{F}_{p^n}[x]$ be a square-free polynomial of degree d.
Let $\mathcal{C}$ be the smooth projective curve with affine model

$$\mathcal{C} : y^r = \overline{F}(x).$$

The zeta function of $\mathcal{C}$ can be computed in time

$$O\left(p^{1/2} \cdot \textit{Polynomial in } n, r, d, \log p\right).$$

```
sage: p = 4999;
sage: x = PolynomialRing(GF(p),"x").gen();
sage: C = CyclicCover(5, x^5 + 1)
sage: C.frobenius_polynomial()
x^12 + 29994*x^10 + 374850015*x^8 + 2498500299980*x^6 + 93675
```

Timings

Our examples were computed on one core of a desktop machine with an `Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz`.

p	time	p	time	p	time
$2^{14}-3$	1.21s	$2^{22}-3$	21.7s	$2^{30}-35$	5m58s
$2^{16}-15$	3.05s	$2^{24}-3$	40.9s	$2^{32}-5$	11m36s
$2^{18}-5$	5.74s	$2^{26}-5$	1m23s	$2^{34}-41$	32m59s
$2^{20}-3$	10.9s	$2^{28}-57$	2m54s	$2^{36}-5$	1h7m

Table: Genus 6 curve $\mathcal{C}\colon y^5 = x^5 - x^4 + x^3 - 2x^2 + 2x + 1$ with $N = 4$

p	time	p	time	p	time
$2^{10}+45$	4m37s	$2^{18}-5$	12m2s	$2^{26}-5$	2h38m
$2^{12}-3$	5m31s	$2^{20}-3$	21m34s	$2^{28}-57$	5h24m
$2^{14}-3$	6m20s	$2^{22}-3$	37m21s	$2^{30}-35$	12h12m
$2^{16}-15$	8m15s	$2^{24}-3$	1h13m	$2^{32}-5$	23h35m

Table: Genus 25 curve $\mathcal{C}\colon y^6 = x^{12} + 10x^{11} + x^{10} + 2x^9 - x^7 - x^5 - 4x^4 + 31x$ with $N = 13$

Timings

Our examples were computed on one core of a desktop machine with an `Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz`.

p	time	p	time	p	time
$2^{12} - 3$	24m1s	$2^{18} - 5$	1h2m	$2^{24} - 3$	7h21m
$2^{14} - 3$	29m50s	$2^{20} - 3$	1h52m	$2^{26} - 5$	16h24m
$2^{16} - 15$	37m14s	$2^{22} - 3$	3h22m	$2^{28} - 57$	33h17m

Table: Genus 45,
$\mathcal{C}\colon y^{11} = x^{11} + 21x^9 + 22x^8 + 12x^7 + 5x^4 + 15x^3 + 6x^2 + 99x + 11$ with $N = 23$