# Computing Riemann–Roch spaces for Algebraic Geometry codes

Elena Berardini

Eindhoven University of Technology

joint with S. Abelard (Thales), A. Couvreur (Inria), G. Lecerf (LIX)

Project funded by the French “Agence de l’Innovation de Défense”

EuroTechPostdoc2 Programme
Excellence in Science and Technology

ACCESS Seminar – 7 June 2022

Click here for the paper

## Linear codes: from Reed–Solomon codes...

Linear code: $\mathbb{F}_q$–vector sub space of $\mathbb{F}_q^n$

$[n, k, d]_q$–code: code of length **n**, dimension **k** and minimum distance **d**

$$\left.\begin{array}{r} \text{dimension} \leftrightarrow \text{information} \\ \text{minimum distance} \leftrightarrow \text{correction capacity} \end{array}\right\} \quad k + d \leqslant n + 1$$

Singleton, 1964

## Linear codes: from Reed–Solomon codes...

Linear code: $\mathbb{F}_q$–vector sub space of $\mathbb{F}_q^n$

$[n,k,d]_q$–code: code of length **n**, dimension **k** and minimum distance **d**

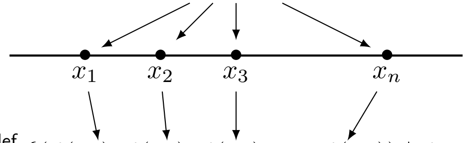
$$\left.\begin{array}{r} \text{dimension} \leftrightarrow \text{information} \\ \text{minimum distance} \leftrightarrow \text{correction capacity} \end{array}\right\} \quad k + d \leqslant n + 1$$

Singleton, 1964

### Reed–Solomon (RS) Codes

Reed and Solomon, 1960

$$f \in \mathbb{F}_q[x]_{<k}$$




$$\mathrm{RS}_k(\mathbf{x}) \stackrel{\text{def}}{=} \{(f(x_1), f(x_2), f(x_3), \ldots, f(x_n)) \mid f \in \mathbb{F}_q[x]_{<k}\}$$

## Linear codes: from Reed–Solomon codes...

Linear code: $\mathbb{F}_q$–vector sub space of $\mathbb{F}_q^n$

$[n,k,d]_q$–code: code of length **n**, dimension **k** and minimum distance **d**

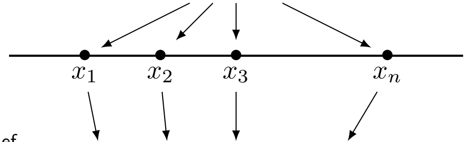
$$\left.\begin{array}{r} \text{dimension} \leftrightarrow \text{information} \\ \text{minimum distance} \leftrightarrow \text{correction capacity} \end{array}\right\} \quad k + d \leqslant n + 1$$

Singleton, 1964

### Reed–Solomon (RS) Codes

Reed and Solomon, 1960

$$f \in \mathbb{F}_q[x]_{<k}$$




$$\mathrm{RS}_k(\mathbf{x}) \stackrel{\text{def}}{=} \{(f(x_1), f(x_2), f(x_3), \dots, f(x_n)) \mid f \in \mathbb{F}_q[x]_{<k}\}$$

- ✔ **Optimal parameters**
  $k + d = n + 1.$
- ✔ **Effective decoding algorithms**
  Berlekamp,1968.

⚠ **Drawback:** $n \leqslant q$.

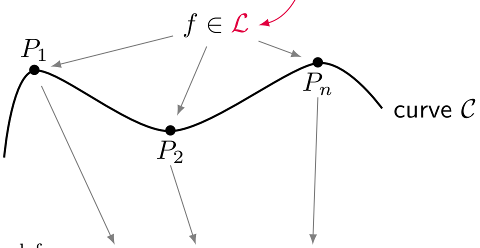
The more $q$ is big,
the less the arithmetic is efficient.

## ...to Algebraic Geometry (AG) codes Goppa, 1981

$$\mathcal{P} = (P_1, P_2, \ldots, P_n)$$

Vector space of functions on the curve

(Riemann–Roch space)



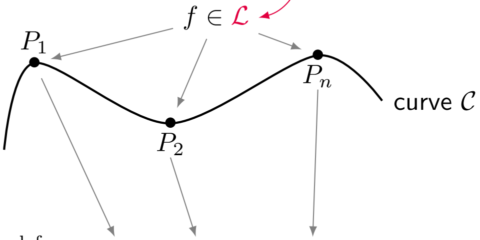

$$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in L\}$$

## ...to Algebraic Geometry (AG) codes Goppa, 1981

$\mathcal{P} = (P_1, P_2, \ldots, P_n)$

Vector space of functions on the curve

(Riemann–Roch space)




$$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in L\}$$

Codes on a **curve** $\mathcal{C}$

- ✔ **Good parameters**
- ✔ **Efficient decoding algorithms**
- ✔ **Length** $> q$

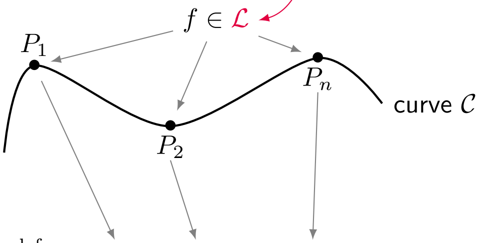
$\#\mathcal{C}(\mathbb{F}_q) \leq q + 1 + g\lfloor 2\sqrt{q} \rfloor$

## ...to Algebraic Geometry (AG) codes Goppa, 1981

$\mathcal{P} = (P_1, P_2, \ldots, P_n)$

Vector space of functions on the curve

(Riemann–Roch space)




$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \overset{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in L\}$

Codes on a **curve** $\mathcal{C}$

- ✔ **Good parameters**
- ✔ **Efficient decoding algorithms**
- ✔ **Length** $> q$

$\#\mathcal{C}(\mathbb{F}_q) \leq q + 1 + g\lfloor 2\sqrt{q} \rfloor$

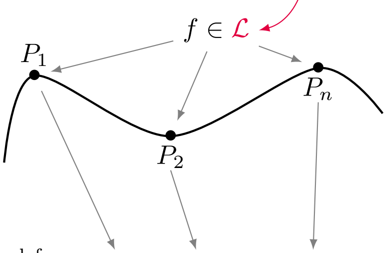
### Proposition

*The parameters* $[n, k, d]$ *of AG codes satisfy* $n + 1 - g \leq k + d \leq n + 1$.

## ...to Algebraic Geometry (AG) codes Goppa, 1981

$\mathcal{P} = (P_1, P_2, \ldots, P_n)$

Vector space of functions on the curve
(Riemann–Roch space)




Codes on a **curve** $\mathcal{C}$

- **Good parameters**
- **Efficient decoding algorithms**
- **Length** $> q$

$\#\mathcal{C}(\mathbb{F}_q) \leq q + 1 + g\lfloor 2\sqrt{q} \rfloor$

$$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in L\}$$

### Proposition

*The parameters* $[n, k, d]$ *of AG codes satisfy* $n + 1 - g \leq k + d \leq n + 1$.

Construction of **good AG codes** relies on $\begin{cases} \text{identify algebraic curves suitable to the context,} \\ \text{design efficient algorithms for implementation.} \end{cases}$

(**identify algebraic curves** suitable to the context, **design efficient algorithms** for implementation.)

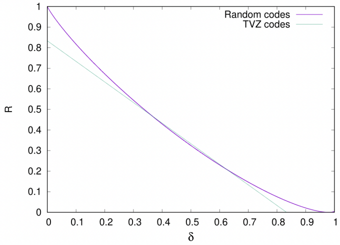
## AG codes: long story short

1981: Goppa introduces AG codes from algebraic curves

## AG codes: long story short

1981: Goppa introduces AG codes from algebraic curves

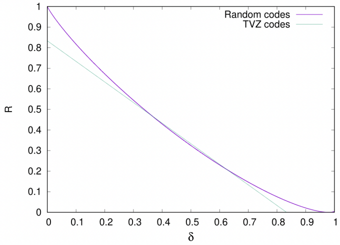
1982: Tsfasman, Vlăduț and Zink use AG codes for beating the Gilbert–Varshamov bound



# AG codes: long story short

1981: Goppa introduces AG codes from algebraic curves

1982: Tsfasman, Vlăduț and Zink use AG codes for beating the Gilbert–Varshamov bound



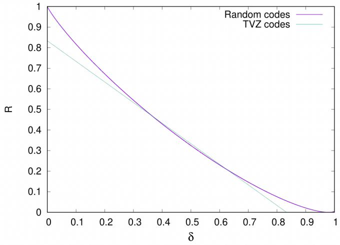

XXc: different familles of curves are studied to obtain good AG codes

↪ the most used curves are the ones for which Riemann–Roch spaces are already known (e.g. Hermitian curves)

## AG codes: long story short

1981: Goppa introduces AG codes from algebraic curves

1982: Tsfasman, Vlăduț and Zink use AG codes for beating the Gilbert–Varshamov bound




XXc: different familles of curves are studied to obtain good AG codes

$\hookrightarrow$ the most used curves are the ones for which Riemann–Roch spaces are already known (e.g. Hermitian curves)

XXIc: AG codes are used in new applications in information theory...

## Riemann–Roch spaces: AG codes and beyond

AG codes provide complexity gains in (not exhaustive list)

- Secret sharing[1]
  Example: can have up to 500 players over $\mathbb{F}_{64}$ with AG codes from maximal curves, while need to work over a field with $> 500$ elements with $\mathrm{RS}$ codes
- Verifiable computing[2]

$\rightsquigarrow$ computing large Riemann–Roch spaces of curves is necessary

[1] R. Cramer, M. Rambaud and C. Xing, Crypto 2021
[2] S. Bordage, M. Lhotel, J. Nardi and H. Randriam, preprint 2022

## Riemann–Roch spaces: AG codes and beyond

AG codes provide complexity gains in (not exhaustive list)

- Secret sharing[1]
  Example: can have up to 500 players over $\mathbb{F}_{64}$ with AG codes from maximal curves, while need to work over a field with $> 500$ elements with $\mathrm{RS}$ codes
- Verifiable computing[2]

$\rightsquigarrow$ computing large Riemann–Roch spaces of curves is necessary

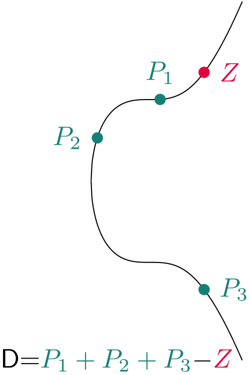
Can be used also for...

- Arithmetic operations on Jacobians of curves[3]
- Symbolic integration[4]

[1] R. Cramer, M. Rambaud and C. Xing, Crypto 2021
[2] S. Bordage, M. Lhotel, J. Nardi and H. Randriam, preprint 2022
[3] K. Khuri-Makdisi, Mathematics of Computations, 2007
[4] J.H. Davenport, Intern. Symp. on Symbolic et Algebraic Manipulation, 1979

## Riemann–Roch spaces of curves

A divisor on a curve $\mathcal{C}$: $D = \sum_{P \in \mathcal{C}} n_P P,\ n_P \in \mathbb{Z}$



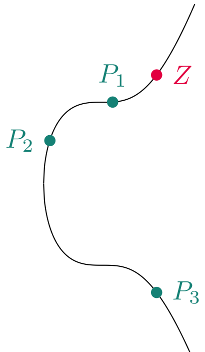

The **Riemann–Roch space** $L(D)$ is the space of functions $\frac{G}{H} \in \mathbb{K}(\mathcal{C})$ such that:

- if $n_P < 0$ then $P$ must be a zero of $G$ (of multiplicity $\geqslant -n_P$)
- if $n_P > 0$ then $P$ can be a zero of $H$ (of multiplicity $\leqslant n_P$)
- $G/H$ has no other poles outside the points $P$ with $n_P > 0$

**Here:** $Z$ must be a zero of $G$, the $P_i$ can be zeros of $H$

## Riemann–Roch spaces of curves

A divisor on a curve $\mathcal{C}$: $D = \sum_{P\in\mathcal{C}} n_P P,\ n_P \in \mathbb{Z}$




D=$P_1 + P_2 + P_3 - Z$

The **Riemann–Roch space** $L(D)$ is the space of functions $\frac{G}{H} \in \mathbb{K}(\mathcal{C})$ such that:

- if $n_P < 0$ then $P$ must be a zero of $G$ (of multiplicity $\geqslant -n_P$)
- if $n_P > 0$ then $P$ can be a zero of $H$ (of multiplicity $\leqslant n_P$)
- $G/H$ has no other poles outside the points $P$ with $n_P > 0$

**Here:** $Z$ must be a zero of $G$, the $P_i$ can be zeros of $H$

**Riemann–Roch Theorem** $\rightsquigarrow$ dimension of $L(D) = \deg D + 1 - g$

where the degree of a divisor is $\deg D = \sum_P n_P \deg(P)$.

## Toy example

Let $\mathcal{C} = \mathbb{P}^1$, $P = [0:1]$ and $Q = [1:1]$. Let $D = P - Q$, then

$$f \in L(D) \iff \begin{cases} \text{f has a zero of order at least } 1 \text{ at } Q, \\ \text{f can have a pole of order at most } 1 \text{ at } P, \\ \text{f has not other poles outside } P. \end{cases}$$

## Toy example

Let $\mathcal{C} = \mathbb{P}^1$, $P = [0:1]$ and $Q = [1:1]$. Let $D = P - Q$, then

$$f \in L(D) \iff \begin{cases} \text{f has a zero of order at least } 1 \text{ at } Q, \\ \text{f can have a pole of order at most } 1 \text{ at } P, \\ \text{f has not other poles outside } P. \end{cases}$$

$f = \frac{X-1}{X}$ is a solution.

## Toy example

Let $\mathcal{C} = \mathbb{P}^1$, $P = [0:1]$ and $Q = [1:1]$. Let $D = P - Q$, then

$$f \in L(D) \iff \begin{cases} \text{f has a zero of order at least } 1 \text{ at } Q, \\ \text{f can have a pole of order at most } 1 \text{ at } P, \\ \text{f has not other poles outside } P. \end{cases}$$

$f = \frac{X-1}{X}$ is a solution.

$$g = 0, \deg D = 0 \xrightarrow[\text{Theorem}]{\text{Riemann–Roch}} \dim L(D) = \deg D + 1 - g = 1$$

$\to f$ generates the space of solutions.

## Toy example

Let $\mathcal{C} = \mathbb{P}^1$, $P = [0:1]$ and $Q = [1:1]$. Let $D = P - Q$, then

$$f \in L(D) \iff \begin{cases} \text{f has a zero of order at least } 1 \text{ at } Q, \\ \text{f can have a pole of order at most } 1 \text{ at } P, \\ \text{f has not other poles outside } P. \end{cases}$$

$f = \frac{X-1}{X}$ is a solution.

$$g = 0, \deg D = 0 \xrightarrow[\text{Theorem}]{\text{Riemann–Roch}} \dim L(D) = \deg D + 1 - g = 1$$

$\rightarrow f$ generates the space of solutions.

⚠ no explicit method to compute a basis of $L(D)$!
How do we solve the problem in general?

## Riemann–Roch problem: state of the art

**Geometric Method:**
(Brill–Noether theory∼1874)
- Goppa, Le Brigand–Risler (80's)
- Huang–Ierardi (90's)
- Khuri–Makdisi (2007)
- Le Gluher–Spaenlehauer (2018)
- Abelard–Couvreur–Lecerf (2020)

**Arithmetic Method:**
(Ideals in function fields)
- Hensel–Landberg (1902)
- Coates (1970)
- Davenport (1981)
- Hess (2001)

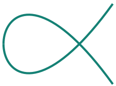
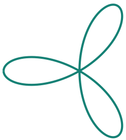
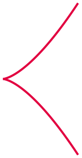
# Riemann–Roch problem: state of the art

**Geometric Method:**
(Brill–Noether theory∼1874)
- Goppa, Le Brigand–Risler (80's)
- Huang–Ierardi (90's)
- Khuri–Makdisi (2007)
- Le Gluher–Spaenlehauer (2018)
- Abelard–Couvreur–Lecerf (2020)

**Arithmetic Method:**
(Ideals in function fields)
- Hensel–Landberg (1902)
- Coates (1970)
- Davenport (1981)
- Hess (2001)

Ordinary/nodal curves: Las Vegas algorithm computing $L(D)$ in sub–quadratic time

Non–ordinary curves: ⚠ no explicit complexity exponent

## Brill–Noether method

Notations:

- $(H) = \sum_{P \in \mathcal{C}} \mathrm{ord}_P(H) P$ – divisor of the zeros of $H$ with multiplicity
- $D \geqslant D' \rightsquigarrow D - D' = \sum n_P P$ with $n_P \geqslant 0 \; \forall P$ ($D - D'$ is *effective*)
  We can always write $D = D_+ - D_-$ with $D_+$ and $D_-$ two effective divisors.

## Brill–Noether method

Notations:

- $(H) = \sum_{P \in \mathcal{C}} \mathrm{ord}_P(H) P$ – divisor of the zeros of $H$ with multiplicity
- $D \geqslant D' \rightsquigarrow D - D' = \sum n_P P$ with $n_P \geqslant 0 \ \forall P$ ($D - D'$ is *effective*)
  We can always write $D = D_+ - D_-$ with $D_+$ and $D_-$ two effective divisors.

**Description of $L(D)$ for $\mathcal{C} : F(X, Y, Z) = 0$ a plane projective curve.**

*The non–zero elements are of the form $\frac{G_i}{H}$ where*

- $H$ *satisfies* $(H) \geqslant D_+$
- $H$ *vanishes at any singular point of* $\mathcal{C}$ *with ad hoc multiplicity*
- $\deg G_i = \deg H$, $G_i$ *prime with* $F$ *and* $(G_i) \geqslant (H) - D$

## Brill–Noether method

Notations:

- $(H) = \sum_{P \in \mathcal{C}} \mathrm{ord}_P(H) P$ – divisor of the zeros of $H$ with multiplicity
- $D \geqslant D' \rightsquigarrow D - D' = \sum n_P P$ with $n_P \geqslant 0 \; \forall P$ ($D - D'$ is *effective*)
  We can always write $D = D_+ - D_-$ with $D_+$ and $D_-$ two effective divisors.

### Description of $L(D)$ for $\mathcal{C} : F(X, Y, Z) = 0$ a plane projective curve.

*The non–zero elements are of the form $\frac{G_i}{H}$ where*

- $H$ *satisfies* $(H) \geqslant D_+$
- $H$ *vanishes at any singular point of* $\mathcal{C}$ *with ad hoc multiplicity*
- $\deg G_i = \deg H$, $G_i$ *prime with* $F$ *and* $(G_i) \geqslant (H) - D$

*How do we manage singular points?*

## Brill–Noether method

Notations:

- $(H) = \sum_{P \in \mathcal{C}} \mathrm{ord}_P(H) P$ – divisor of the zeros of $H$ with multiplicity
- $D \geqslant D' \rightsquigarrow D - D' = \sum n_P P$ with $n_P \geqslant 0 \; \forall P$ ($D - D'$ is *effective*)
  We can always write $D = D_+ - D_-$ with $D_+$ and $D_-$ two effective divisors.

### Description of $L(D)$ for $\mathcal{C} : F(X, Y, Z) = 0$ a plane projective curve.

*The non–zero elements are of the form* $\frac{G_i}{H}$ *where*

- $H$ *satisfies* $(H) \geqslant D_+$
- $H$ *vanishes at any singular point of* $\mathcal{C}$ *with ad hoc multiplicity*
- $\deg G_i = \deg H$, $G_i$ *prime with* $F$ *and* $(G_i) \geqslant (H) - D$

*How do we manage singular points?*

the adjoint divisor $\mathcal{A}$ "encodes" the singular points of $\mathcal{C}$ with their multiplicities

## Brill–Noether method

Notations:

- $(H) = \sum_{P \in \mathcal{C}} \mathrm{ord}_P(H)P$ – divisor of the zeros of $H$ with multiplicity
- $D \geqslant D' \rightsquigarrow D - D' = \sum n_P P$ with $n_P \geqslant 0\ \forall P$ ($D - D'$ is *effective*)

  We can always write $D = D_+ - D_-$ with $D_+$ and $D_-$ two effective divisors.

**Description of $L(D)$ for $\mathcal{C} : F(X, Y, Z) = 0$ a plane projective curve.**

*The non–zero elements are of the form $\frac{G_i}{H}$ where*

- *$H$ satisfies $(H) \geqslant D_+$*
- *$H$ satisfies $(H) \geqslant \mathcal{A}$ (we say that "$H$ is adjoint to the curve")*
- $\deg G_i = \deg H$, *$G_i$ prime with $F$ and $(G_i) \geqslant (H) - D$*

*How do we manage singular points?*

the adjoint divisor $\mathcal{A}$ "encodes" the singular points of $\mathcal{C}$ with their multiplicities

## Brill–Noether method

Notations:

- $(H) = \sum_{P \in \mathcal{C}} \mathrm{ord}_P(H) P$ – divisor of the zeros of $H$ with multiplicity
- $D \geqslant D' \rightsquigarrow D - D' = \sum n_P P$ with $n_P \geqslant 0\ \forall P$ ($D - D'$ is *effective*)
  We can always write $D = D_+ - D_-$ with $D_+$ and $D_-$ two effective divisors.

**Description of $L(D)$ for $\mathcal{C} : F(X, Y, Z) = 0$ a plane projective curve.**

*The non–zero elements are of the form $\frac{G_i}{H}$ where*

- *$H$ satisfies $(H) \geqslant D_+$*
- *$H$ satisfies $(H) \geqslant \mathcal{A}$*
- $\deg G_i = \deg H$, *$G_i$ prime with $F$ and $(G_i) \geqslant (H) - D$*

*How do we manage singular points?*

the adjoint divisor $\mathcal{A}$ "encodes" the singular points of $\mathcal{C}$ with their multiplicities

*How do we represent divisors?*

## Brill–Noether method

Notations:

- $(H) = \sum_{P \in \mathcal{C}} \mathrm{ord}_P(H)P$ – divisor of the zeros of $H$ with multiplicity
- $D \geqslant D' \rightsquigarrow D - D' = \sum n_P P$ with $n_P \geqslant 0 \ \forall P$ ($D - D'$ is *effective*)

  We can always write $D = D_+ - D_-$ with $D_+$ and $D_-$ two effective divisors.

**Description of $L(D)$ for $\mathcal{C} : F(X, Y, Z) = 0$ a plane projective curve.**

*The non–zero elements are of the form $\frac{G_i}{H}$ where*

- *$H$ satisfies $(H) \geqslant D_+$*
- *$H$ satisfies $(H) \geqslant \mathcal{A}$*
- $\deg G_i = \deg H$, *$G_i$ prime with $F$ and $(G_i) \geqslant (H) - D$*

*How do we manage singular points?*

the adjoint divisor $\mathcal{A}$ "encodes" the singular points of $\mathcal{C}$ with their multiplicities

*How do we represent divisors?*

series expansions of multi–set $((P_i)_i, n_i)$ $\quad\rightarrow\quad$ operations on divisors with negligible cost

## Sketch of the algorithm

**Input**

*$\mathcal{C} : F(X, Y, Z) = 0$ a plane curve of degree $\delta$, $D$ a smooth divisor.*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$

**Step 2 :** Compute the common denominator $H$

**Step 3 :** Compute $(H) - D$

**Step 4 :** Compute the numerators $G_i$ (similar to Step 2)

**Output**

*A basis of the Riemann–Roch space $L(D)$ in terms of $H$ and the $G_i$.*

## Sketch of the algorithm

### Input

$\mathcal{C} : F(X, Y, Z) = 0$ *a plane curve of degree* $\delta$, $D$ *a smooth divisor.*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$

**Step 2 :** Compute the common denominator $H$

**Step 3 :** Compute $(H) - D$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^2)$

**Step 4 :** Compute the numerators $G_i$ (similar to Step 2)

### Output

*A basis of the Riemann–Roch space* $L(D)$ *in terms of* $H$ *and the* $G_i$*.*

## Sketch of the algorithm

### Input

$\mathcal{C} : F(X, Y, Z) = 0$ *a plane curve of degree* $\delta$, $D$ *a smooth divisor.*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$

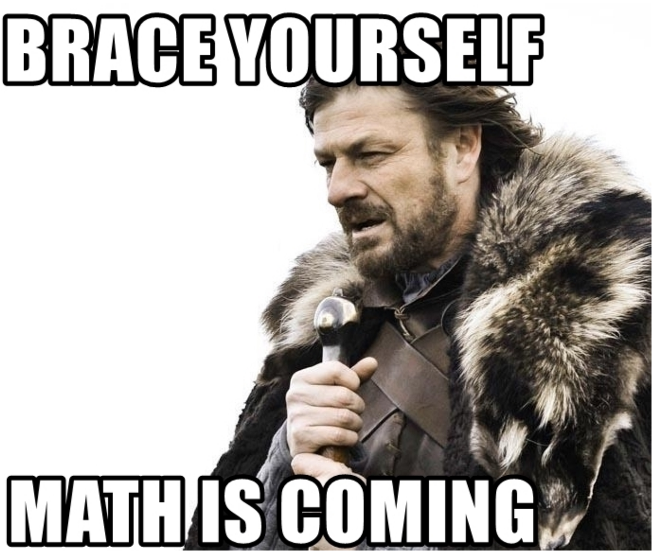
**Step 2 :** Compute the common denominator $H$

**Step 3 :** Compute $(H) - D$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^2)$

**Step 4 :** Compute the numerators $G_i$ (similar to Step 2)

### Output

*A basis of the Riemann–Roch space* $L(D)$ *in terms of* $H$ *and the* $G_i$*.*


BRACE YOURSELF
MATH IS COMING

## Warm up: adjoint divisor in the ordinary case

### Definition

*Let $\mathcal{C}$ be defined over a field $\mathbb{K}$, and let $P \in \mathrm{Sing}(\mathcal{C})$. The local adjoint divisor is*

$$\mathcal{A}_P = -\sum_{\mathcal{P}|P} \mathrm{val}_{\mathcal{P}}\left(\frac{dx}{F_y(x,y,1)}\right)\mathcal{P}.$$

## Warm up: adjoint divisor in the ordinary case

### Definition

*Let $\mathcal{C}$ be defined over a field $\mathbb{K}$, and let $P \in \mathrm{Sing}(\mathcal{C})$. The local adjoint divisor is*

$$\mathcal{A}_P = -\sum_{\mathcal{P}|P} \mathrm{val}_{\mathcal{P}} \left( \frac{dx}{F_y(x,y,1)} \right) \mathcal{P}.$$

Let $P \in \mathrm{Sing}(\mathcal{C})$ ordinary of multiplicity $m$, wlog $P = (0:0:1)$. Then $F$ locally factorises as

$$F(x,y,1) = u(x,y) \prod_{i=1}^{m} (y - \varphi_i(x))$$

with $u \in \overline{\mathbb{K}}[[x,y]]$ invertible, $\varphi_i(x) \in x\overline{\mathbb{K}}[[x]]$ and $\varphi_i'(0) \neq \varphi_j'(0)$.

## Warm up: adjoint divisor in the ordinary case

### Definition

*Let $\mathcal{C}$ be defined over a field $\mathbb{K}$, and let $P \in \mathrm{Sing}(\mathcal{C})$. The local adjoint divisor is*

$$\mathcal{A}_P = -\sum_{\mathcal{P}|P} \mathrm{val}_{\mathcal{P}}\left(\frac{dx}{F_y(x,y,1)}\right)\mathcal{P}.$$

Let $P \in \mathrm{Sing}(\mathcal{C})$ ordinary of multiplicity $m$, wlog $P = (0:0:1)$. Then $F$ locally factorises as

$$F(x,y,1) = u(x,y)\prod_{i=1}^{m}(y - \varphi_i(x))$$

with $u \in \overline{\mathbb{K}}[[x,y]]$ invertible, $\varphi_i(x) \in x\overline{\mathbb{K}}[[x]]$ and $\varphi_i'(0) \neq \varphi_j'(0)$.

Germ of the curve parametrized by $\varphi_i(x)$ $\longleftrightarrow$ place $\mathcal{P}_i$ in the functions field $\overline{\mathbb{K}}(\mathcal{C})$

## Warm up: adjoint divisor in the ordinary case

### Definition

*Let $\mathcal{C}$ be defined over a field $\mathbb{K}$, and let $P \in \mathrm{Sing}(\mathcal{C})$. The local adjoint divisor is*

$$\mathcal{A}_P = -\sum_{\mathcal{P}|P} \mathrm{val}_{\mathcal{P}}\left(\frac{dx}{F_y(x,y,1)}\right)\mathcal{P}.$$

Let $P \in \mathrm{Sing}(\mathcal{C})$ ordinary of multiplicity $m$, wlog $P = (0:0:1)$. Then $F$ locally factorises as

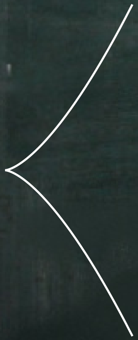
$$F(x,y,1) = u(x,y)\prod_{i=1}^{m}(y - \varphi_i(x))$$

with $u \in \overline{\mathbb{K}}[[x,y]]$ invertible, $\varphi_i(x) \in x\overline{\mathbb{K}}[[x]]$ and $\varphi_i'(0) \neq \varphi_j'(0)$.

Germ of the curve parametrized by $\varphi_i(x)$ $\longleftrightarrow$ place $\mathcal{P}_i$ in the functions field $\overline{\mathbb{K}}(\mathcal{C})$

The *local adjoint divisor* becomes $\mathcal{A}_P = (m-1)\sum_{i=1}^{m}\mathcal{P}_i$.

## Example



$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

# Example

$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0,0)$ non–ordinary singular point of multiplicity $2$

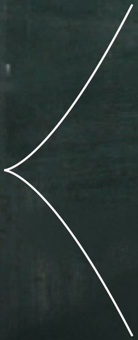
## Example

$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0,0)$ non–ordinary singular point of multiplicity $2$

"Factorisation": $(y - x^{3/2})(y + x^{3/2}) = 0$

## Example



$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0, 0)$ non–ordinary singular point of multiplicity $2$

"Factorisation": $(y - x^{3/2})(y + x^{3/2}) = 0$

We use Puiseux series!

## Adjoint condition via Puiseux series

Informally: Puiseux series are Laurent series that admit fractional exponents.

$F \in \mathbb{K}((x))[y]$ has $\deg F = d$ distinct roots in its field of Puiseux series and writes as

$$F = \prod_{i=1}^{d} (y - \varphi_i) = \prod_{i=1}^{d} \left( y - \sum_{j=n}^{\infty} \beta_{i,j} x^{j/e_i} \right).$$

## Adjoint condition via Puiseux series

Informally: Puiseux series are Laurent series that admit fractional exponents.

$F \in \mathbb{K}((x))[y]$ has $\deg F = d$ distinct roots in its field of Puiseux series and writes as

$$F = \prod_{i=1}^{d} (y - \varphi_i) = \prod_{i=1}^{d} \left( y - \sum_{j=n}^{\infty} \beta_{i,j} x^{j/e_i} \right).$$

We fix $\varphi$ of degree $e$, $\zeta$ a primitive $e$-th root of unity. For $0 \leqslant k < e$ we can construct other $e$ Puiseux series by replacing $x^{1/e}$ with $\zeta^k x^{1/e}$.

## Adjoint condition via Puiseux series

Informally: Puiseux series are Laurent series that admit fractional exponents.

$F \in \mathbb{K}((x))[y]$ has $\deg F = d$ distinct roots in its field of Puiseux series and writes as

$$F = \prod_{i=1}^{d} (y - \varphi_i) = \prod_{i=1}^{d} \left( y - \sum_{j=n}^{\infty} \beta_{i,j} x^{j/e_i} \right).$$

We fix $\varphi$ of degree $e$, $\zeta$ a primitive $e$-th root of unity. For $0 \leqslant k < e$ we can construct other $e$ Puiseux series by replacing $x^{1/e}$ with $\zeta^k x^{1/e}$. They are all equivalent and represented by...

### Definition

*A Rational Puiseux Expansion (RPE) is a pair* $(X(t), Y(t)) = \left( \gamma t^e, \sum_{j=n}^{\infty} \beta_j t^j \right)$ *such that* $F(X(t), Y(t)) = 0$.

## Adjoint condition via Puiseux series

Informally: Puiseux series are Laurent series that admit fractional exponents.

$F \in \mathbb{K}((x))[y]$ has $\deg F = d$ distinct roots in its field of Puiseux series and writes as

$$F = \prod_{i=1}^{d} (y - \varphi_i) = \prod_{i=1}^{d} \left( y - \sum_{j=n}^{\infty} \beta_{i,j} x^{j/e_i} \right).$$

We fix $\varphi$ of degree $e$, $\zeta$ a primitive $e$-th root of unity. For $0 \leqslant k < e$ we can construct other $e$ Puiseux series by replacing $x^{1/e}$ with $\zeta^k x^{1/e}$. They are all equivalent and represented by...

### Definition

*A* Rational Puiseux Expansion (RPE) *is a pair* $(X(t), Y(t)) = \left(\gamma t^e, \sum_{j=n}^{\infty} \beta_j t^j\right)$ *such that* $F(X(t), Y(t)) = 0$.

Rational Puiseux Expansion of $F(x, y, 1)$ $\longleftrightarrow$ places of $\overline{\mathbb{K}}(\mathcal{C})$ in the chart $z = 1$

## The adjoint divisor

Let $P \in \mathrm{Sing}(\mathcal{C})$ ~~ordinary~~, w.l.o.g. $P = (0:0:1)$. Then $F$ locally factorises as

$$F(x,y,1) = u(x,y)\prod_{i=1}^{m}(y - \varphi_i(x)),$$

with $u \in \mathbb{K}[[x,y]]$ invertible and $\varphi_i$ Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

## The adjoint divisor

Let $P \in \mathrm{Sing}(\mathcal{C})$ ~~ordinary~~, w.l.o.g. $P = (0:0:1)$. Then $F$ locally factorises as

$$F(x,y,1) = u(x,y)\prod_{i=1}^{m}(y-\varphi_i(x)),$$

with $u \in \mathbb{K}[[x,y]]$ invertible and $\varphi_i$ Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

$$\{\varphi_1,\dots,\varphi_m\} \qquad \rightsquigarrow \qquad \begin{array}{c}\text{RPEs/places } (X_i(t),Y_i(t)) \\ i\in\{1,\dots,s\},\ s\leqslant m.\end{array}$$

## The adjoint divisor

Let $P \in \mathrm{Sing}(\mathcal{C})$ ~~ordinary~~, w.l.o.g. $P = (0:0:1)$. Then $F$ locally factorises as

$$F(x,y,1) = u(x,y) \prod_{i=1}^{m} (y - \varphi_i(x)),$$

with $u \in \mathbb{K}[[x,y]]$ invertible and $\varphi_i$ Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

$$\{\varphi_1, \ldots, \varphi_m\} \quad \rightsquigarrow \quad \begin{array}{c}\text{RPEs/places } (X_i(t), Y_i(t)) \\ i \in \{1, \ldots, s\},\ s \leqslant m.\end{array}$$

The local adjoint divisor becomes

$$\mathcal{A}_P = -\sum_{\mathcal{P}|P} \mathrm{val}_t \left( \frac{e t^{e-1}}{F_y(X(t), Y(t), 1)} \right) \mathcal{P}.$$

## The adjoint divisor

Let $P \in \operatorname{Sing}(\mathcal{C})$ ~~ordinary~~, w.l.o.g. $P = (0:0:1)$. Then $F$ locally factorises as

$$F(x, y, 1) = u(x, y) \prod_{i=1}^{m} (y - \varphi_i(x)),$$

with $u \in \mathbb{K}[[x, y]]$ invertible and $\varphi_i$ Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

$\{\varphi_1, \dots, \varphi_m\} \quad \rightsquigarrow \quad$ RPEs/places $(X_i(t), Y_i(t))$
$i \in \{1, \dots, s\}$, $s \leqslant m$.

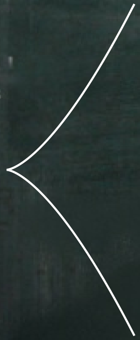
The local adjoint divisor becomes

$$\mathcal{A}_P = -\sum_{\mathcal{P}|P} \operatorname{val}_t \left( \frac{e t^{e-1}}{F_y(X(t), Y(t), 1)} \right) \mathcal{P}.$$

**In practice:** algorithm for computing Puiseux series[5] $\rightsquigarrow$ $\mathcal{A}$ computed with $\tilde{O}(\delta^3)$ operations.

[5] A. Poteaux and M. Weimann, Annales Herni Lebesgue, 2021

# Example

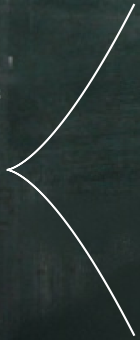


$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0,0)$ unique singular point, non–ordinary

Puiseux series: $(y - x^{3/2})(y + x^{3/2}) = 0$

# Example



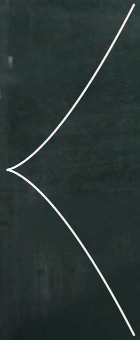
$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0,0)$ unique singular point, non–ordinary

Puiseux series: $(y - x^{3/2})(y + x^{3/2}) = 0$

(Unique) RPE: $(X(t), Y(t)) = (t^2, t^3)$

## Example



$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0,0)$ unique singular point, non–ordinary

Puiseux series: $(y - x^{3/2})(y + x^{3/2}) = 0$

(Unique) RPE: $(X(t), Y(t)) = (t^2, t^3)$

Adjoint condition: $F_y = 2y$, $x = t^2 \Rightarrow dx = 2t$

# Example

$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0,0)$ unique singular point, non–ordinary

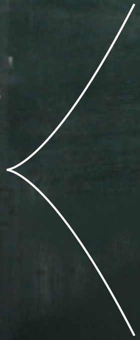
Puiseux series: $(y - x^{3/2})(y + x^{3/2}) = 0$

(Unique) RPE: $(X(t), Y(t)) = (t^2, t^3)$

Adjoint condition: $F_y = 2y$, $x = t^2 \Rightarrow dx = 2t$

$$\mathrm{val}_t \left( \frac{e t^{e-1}}{F_y(X(t),Y(t),1)} \right) = \mathrm{val}_t \left( \frac{2t}{2t^3} \right) = \mathrm{val}_t \left( \frac{1}{t^2} \right) = -2$$

## Example



$\mathcal{C} : y^2 - x^3 = 0$ in the chart $z = 1$

$(0, 0)$ unique singular point, non–ordinary

Puiseux series: $(y - x^{3/2})(y + x^{3/2}) = 0$

(Unique) RPE: $(X(t), Y(t)) = (t^2, t^3)$

Adjoint condition: $F_y = 2y$, $x = t^2 \Rightarrow dx = 2t$

$$\operatorname{val}_t\left(\frac{et^{e-1}}{F_y(X(t),Y(t),1)}\right) = \operatorname{val}_t\left(\frac{2t}{2t^3}\right) = \operatorname{val}_t\left(\frac{1}{t^2}\right) = -2$$

$$(H) \geq \mathcal{A} \iff \operatorname{val}_t H(t^2, t^3) \geq 2$$

## Sketch of the algorithm

### Input

$\mathcal{C} : F(X, Y, Z) = 0$ *a plane curve of degree* $\delta$, $D$ *a smooth divisor .*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$ ✔ $\leftarrow \tilde{O}(\delta^3)$

**Step 2 :** Compute the common denominator $H$

**Step 3 :** Compute $(H) - D \leftarrow \tilde{O}((\delta^2 + \deg D_+)^2)$

**Step 4 :** Compute the numerators $G_i$ (similar to Step 2)

### Output

*A basis of the Riemann–Roch space* $L(D)$ *in terms of* $H$ *and the* $G_i$.

## Sketch of the algorithm

### Input

$\mathcal{C} : F(X, Y, Z) = 0$ *a plane curve of degree* $\delta$, $D$ *a smooth divisor .*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$ ✔ $\leftarrow \tilde{O}(\delta^3)$

**Step 2 :** Compute the common denominator $H$

**Step 3 :** Compute $(H) - D \leftarrow \tilde{O}((\delta^2 + \deg D_+)^2)$

**Step 4 :** Compute the numerators $G_i$ (similar to Step 2)

### Output

*A basis of the Riemann–Roch space* $L(D)$ *in terms of* $H$ *and the* $G_i$.

## Find a denominator in practice: classical linear algebra

Let $d := \deg H$.

$$\text{Condition } (H) \geqslant \mathcal{A} + D_+$$

$\rightsquigarrow$ linear system with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

$\rightsquigarrow$ we retrieve $H$ by Gauss elimination that costs

$$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega) \text{ operations}^6 \text{ in } \mathbb{K}.$$

[6] $2 \leqslant \omega \leqslant 3$ is a feasible exponent for linear algebra ($\omega = 2.373$)

## Find a denominator in practice: classical linear algebra

Let $d := \deg H$.

Condition $(H) \geqslant \mathcal{A} + D_+$

$\rightsquigarrow$ linear system with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

$\rightsquigarrow$ we retrieve $H$ by Gauss elimination that costs

$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega)$ operations[6] in $\mathbb{K}$.

**How big is $d$?**

We showed that $d = \left\lceil \frac{(\delta-1)(\delta-2)+\deg D_+}{\delta} \right\rceil$ is enough

$\rightsquigarrow$ denominator computed with $\tilde{O}((\delta^2 + \deg D_+)^\omega)$ operations in $\mathbb{K}$.

[6] $2 \leqslant \omega \leqslant 3$ is a feasible exponent for linear algebra ($\omega = 2.373$)

## Second method: structured linear algebra

Condition $(H) \geqslant \mathcal{A}$

$$\rightsquigarrow \mathrm{val}_t(H(X(t), Y(t), 1) \geqslant -\mathrm{val}_t \left( \frac{e t^{e-1}}{F_y(X(t), Y(t), 1)} \right)$$

(similar equations for the condition $(H) \geqslant D_+$ )

The space of polynomials $H(x, y, 1)$ that satisfy these conditions is a $\mathbb{K}[x]$–module

$\rightsquigarrow$ computing a basis[7] costs $\tilde{O}((\delta^2 + \deg D)^\omega)$ operations in $\mathbb{K}$.

[7] C.-P. Jeannerod, V. Neiger, É. Schost and G. Villard, J. Symbolic Comput. 2017

## Second method: structured linear algebra

Condition $(H) \geqslant \mathcal{A}$

$$\rightsquigarrow \mathrm{val}_t(H(X(t), Y(t), 1) \geqslant -\mathrm{val}_t\left(\frac{et^{e-1}}{F_y(X(t), Y(t), 1)}\right)$$

(similar equations for the condition $(H) \geqslant D_+$ )

The space of polynomials $H(x, y, 1)$ that satisfy these conditions is a $\mathbb{K}[x]$–module

$\rightsquigarrow$ computing a basis[7] costs $\tilde{O}((\delta^2 + \deg D)^{\omega})$ operations in $\mathbb{K}$.

Same complexity exponent but with some

Advantages:

- better complexity exponent over algebraically closed fields: $\tilde{O}((\delta^2 + \deg D)^{\frac{\omega+1}{2}})$,
- potential improvement in the future.

[7] C.-P. Jeannerod, V. Neiger, É. Schost and G. Villard, J. Symbolic Comput. 2017

## Sketch of the algorithm

### Input

$\mathcal{C} : F(X,Y,Z) = 0$ *a plane curve of degree* $\delta$, $D$ *a smooth divisor .*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$ ✔ $\leftarrow \tilde{O}(\delta^3)$

**Step 2 :** Compute the common denominator $H$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^\omega)$

**Step 3 :** Compute $(H) - D$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^2)$

**Step 4 :** Compute the numerators $G_i$ (similar to Step 2)

### Output

*A basis of the Riemann–Roch space* $L(D)$ *in terms of* $H$ *and the* $G_i$.

## Sketch of the algorithm

### Input

$\mathcal{C} : F(X,Y,Z) = 0$ *a plane curve of degree* $\delta$, $D$ *a smooth divisor .*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$ ✔ $\leftarrow \tilde{O}(\delta^3)$

**Step 2 :** Compute the common denominator $H$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^\omega)$

**Step 3 :** Compute $(H) - D$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^2)$

**Step 4 :** Compute the numerators $G_i$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^\omega)$

### Output

*A basis of the Riemann–Roch space* $L(D)$ *in terms of* $H$ *and the* $G_i$*.*

## Sketch of the algorithm

### Input

*$\mathcal{C} : F(X,Y,Z) = 0$ a plane curve of degree $\delta$, $D$ a smooth divisor .*

**Step 1 :** Compute the adjoint divisor $\mathcal{A}$ ✔ $\leftarrow \tilde{O}(\delta^3)$

**Step 2 :** Compute the common denominator $H$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^\omega)$

**Step 3 :** Compute $(H) - D$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^2)$

**Step 4 :** Compute the numerators $G_i$ ✔ $\leftarrow \tilde{O}((\delta^2 + \deg D_+)^\omega)$

### Output

*A basis of the Riemann–Roch space $L(D)$ in terms of $H$ and the $G_i$.*

### Theorem (Abelard, B–, Couvreur, Lecerf – Journal of Complexity 2022)

*The previous algorithm computes $L(D)$ with $\tilde{O}((\delta^2 + \deg D_+)^\omega)$ operations in $\mathbb{K}$.*

## What to take away?

0. Implementation of AG codes $\rightsquigarrow$ need to compute large Riemann–Roch spaces $L(D)$
1. Brill–Noether method $\rightsquigarrow$ necessary and sufficient conditions on $G$ and $H$ such that $G/H \in L(D)$
2. Puiseux series $\rightsquigarrow$ management of *non–ordinary* singular points of the curve
3. Linear Algebra $\rightsquigarrow$ Computing $H$ and $G$ in practice

## What to take away?

0. Implementation of AG codes $\rightsquigarrow$ need to compute large Riemann–Roch spaces $L(D)$
1. Brill–Noether method $\rightsquigarrow$ necessary and sufficient conditions on $G$ and $H$ such that $G/H \in L(D)$
2. Puiseux series $\rightsquigarrow$ management of *non–ordinary* singular points of the curve
3. Linear Algebra $\rightsquigarrow$ Computing $H$ and $G$ in practice

**Main result**

*We can compute Riemann–Roch spaces of any plane curve with a good complexity exponent.*

## Future questions

- Computing Riemann–Roch spaces of non–ordinary curves in positive “small” characteristic (in progress).

  **Main obstacle:** find an alternative tool to Puiseux series to handle the adjoint condition.

## Future questions

- Computing Riemann–Roch spaces of non–ordinary curves in positive "small" characteristic (in progress).

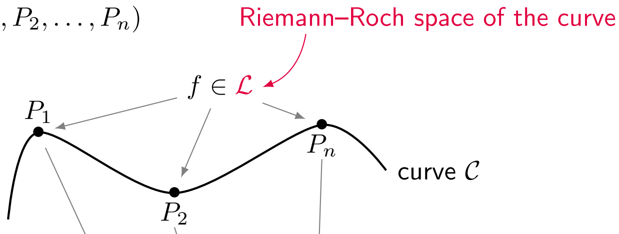
  **Main obstacle:** find an alternative tool to Puiseux series to handle the adjoint condition.

- Improving the complexity exponent in the non–ordinary case. (Sub–quadratic as in the ordinary case?)

  **Main obstacle:** linear algebra.

## AG codes: from curves to surfaces

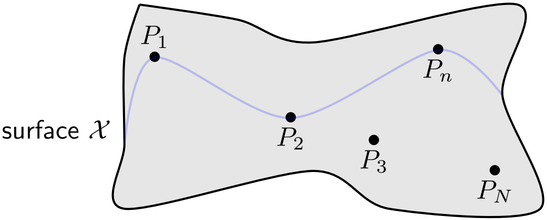
$\mathcal{P} = (P_1, P_2, \ldots, P_n)$




$$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in \mathcal{L}\}$$

## AG codes: from curves to surfaces

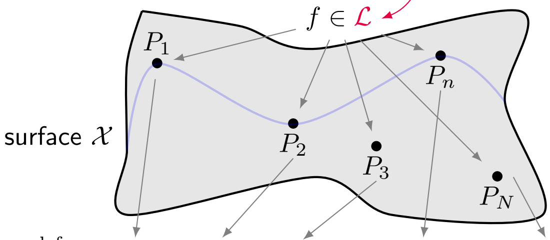
$$\mathcal{P} = (P_1, P_2, P_3, \ldots, P_n, \ldots, P_N)$$



## AG codes: from curves to surfaces

$$\mathcal{P} = (P_1, P_2, P_3, \ldots, P_n, \ldots, P_N)$$

Riemann–Roch space of the **surface**



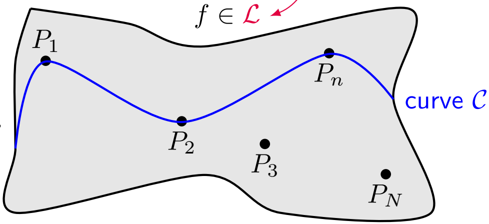

$$C_{\mathcal{X}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), f(P_3), \ldots, f(P_n), \ldots, f(P_N)) \mid f \in \mathcal{L}\}$$

Codes on a **surface** $\mathcal{X}$

- ✔ **Length:** $N \sim q^2$
- **??** **Parameters & decoding**
  (✔ **very** particular cases)

## AG codes: from curves to surfaces

$\mathcal{P} = (P_1, P_2, P_3, \ldots, P_n, \ldots, P_N)$ Riemann–Roch space of the **surface**




Codes on a **surface** $\mathcal{X}$

- ✔ **Length:** $N \sim q^2$
- **??** **Parameters & decoding**
  (✔ **very** particular cases)

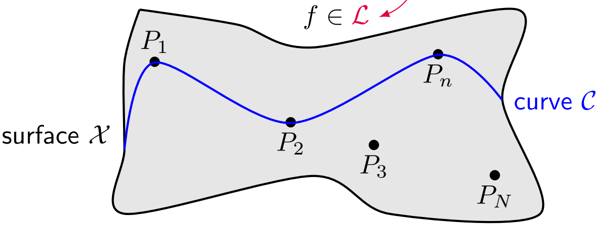
$$C_{\mathcal{X}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), f(P_3), \ldots, f(P_n), \ldots, f(P_N)) \mid f \in \mathcal{L}\}$$

Restriction to $\mathcal{C}$

$$\{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in \mathcal{L}\}$$

## AG codes: from curves to surfaces

$\mathcal{P} = (P_1, P_2, P_3, \ldots, P_n, \ldots, P_N)$ Riemann–Roch space of the **surface**




Codes on a **surface** $\mathcal{X}$

- ✔ **Length:** $N \sim q^2$
- **??** **Parameters & decoding**
  (✔ **very** particular cases)

$$C_{\mathcal{X}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), f(P_3), \ldots, f(P_n), \ldots, f(P_N)) \mid f \in \mathcal{L}\}$$

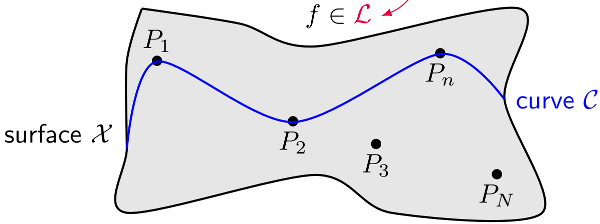
Restriction to $\mathcal{C}$

$$C_{\mathcal{C}}(\mathcal{L}_{\mathcal{C}}, \mathcal{P}_{\mathcal{C}}) = \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in \mathcal{L}\}$$

✔ Local properties from curves lying on $\mathcal{X}$
(*e.g.* local decoding, local recoverability)

## AG codes: from curves to surfaces

$\mathcal{P} = (P_1, P_2, P_3, \dots, P_n, \dots, P_N)$ Riemann–Roch space of the **surface**




Codes on a **surface** $\mathcal{X}$

- ✔ **Length:** $N \sim q^2$
- **??** **Parameters & decoding**
  (✔ **very** particular cases)

$$C_{\mathcal{X}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), f(P_3), \dots, f(P_n), \dots, f(P_N)) \mid f \in \mathcal{L}\}$$

Restriction to $\mathcal{C}$

$$C_{\mathcal{C}}(\mathcal{L}_{\mathcal{C}}, \mathcal{P}_{\mathcal{C}}) = \{(f(P_1), f(P_2), \dots, f(P_n)) \mid f \in \mathcal{L}\}$$

✔ Local properties from curves lying on $\mathcal{X}$
(*e.g.* local decoding, local recoverability)

- Can we develop a "Brill–Noether" theory for computing Riemann–Roch spaces of surfaces?

# Thank you for your attention!

Questions?

e.berardini@tue.nl