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"

Contemporary and Geometric Techniques in Coding Theory and Cryptography
18 July 2022

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...

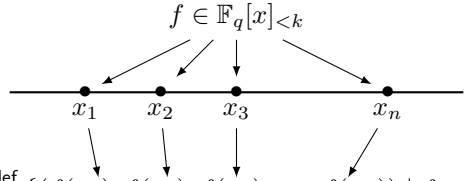
Linear code: $\mathbb{F}_q$–vector sub space of $\mathbb{F}_q^n$

$[n, k, d]_q$–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



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

Linear codes: from Reed–Solomon codes...

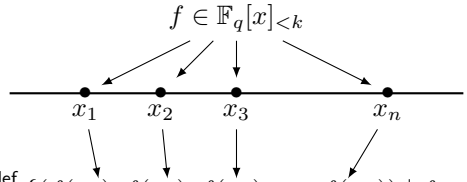
Linear code: $\mathbb{F}_q$–vector sub space of $\mathbb{F}_q^n$

$[n, k, d]_q$–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



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

- ✔ **Optimal parameters**
  $k + d = n + 1$.
- ✔ **Effective decoding algorithms**
  Berlekamp,1968.
- ⚠ **Drawback:** $n \leqslant q$.

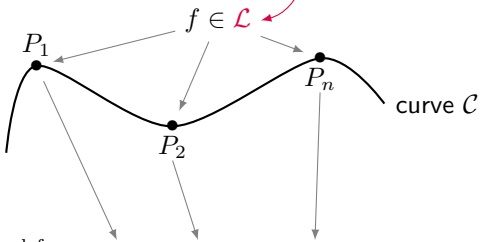
The bigger the q,
the less efficient the arithmetic.

...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)



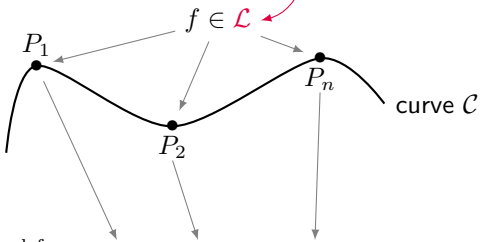
$$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in L\}$$

...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** $> q$

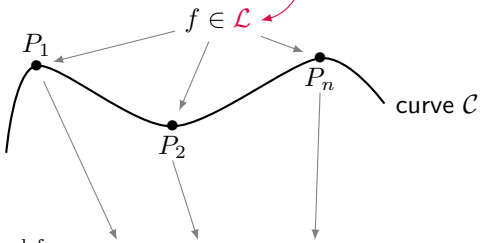
$\#\mathcal{C}(\mathbb{F}_q) \leq q + 1 + g\lfloor 2\sqrt{q} \rfloor$

...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)



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

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

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

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

Proposition

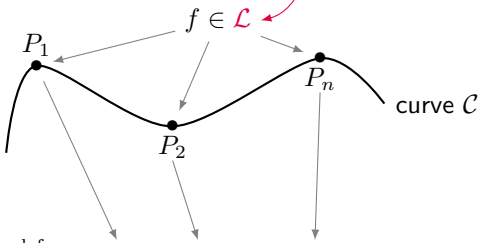
The parameters $[n, k, d]$ *of AG codes satisfy* $n + 1 - g \leq k + d \leq n + 1$.

...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)



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

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

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

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

Proposition

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

AG codes are at distance g *from optimality*

AG codes: applications and construction

AG codes found application in (not exhaustive list)

- Constructing quantum error correcting codes[1]
  Necessary for the implementation of quantum computers.
- Secret sharing[2]
  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 RS codes.
- Verifiable computing[3]
  For zero–knowledge proofs, blockchain...

[1] La Guardia, Pereira, Quantum Information Processing, 2017
[2] R. Cramer, M. Rambaud and C. Xing, Crypto 2021
[3] S. Bordage, M. Lhotel, J. Nardi and H. Randriam, CCC 2022

AG codes: applications and construction

AG codes found application in (not exhaustive list)

- Constructing quantum error correcting codes[1]
  Necessary for the implementation of quantum computers.
- Secret sharing[2]
  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 RS codes.
- Verifiable computing[3]
  For zero–knowledge proofs, blockchain...

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

[1] La Guardia, Pereira, Quantum Information Processing, 2017
[2] R. Cramer, M. Rambaud and C. Xing, Crypto 2021
[3] S. Bordage, M. Lhotel, J. Nardi and H. Randriam, CCC 2022

AG codes: applications and construction

AG codes found application in (not exhaustive list)

- Constructing quantum error correcting codes[1]
  Necessary for the implementation of quantum computers.
- Secret sharing[2]
  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 RS codes.
- Verifiable computing[3]
  For zero–knowledge proofs, blockchain...

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

[1] La Guardia, Pereira, Quantum Information Processing, 2017
[2] R. Cramer, M. Rambaud and C. Xing, Crypto 2021
[3] S. Bordage, M. Lhotel, J. Nardi and H. Randriam, CCC 2022

AG codes: applications and construction

AG codes found application in (not exhaustive list)

- Constructing quantum error correcting codes[1]
  Necessary for the implementation of quantum computers.
- Secret sharing[2]
  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 RS codes.
- Verifiable computing[3]
  For zero–knowledge proofs, blockchain...

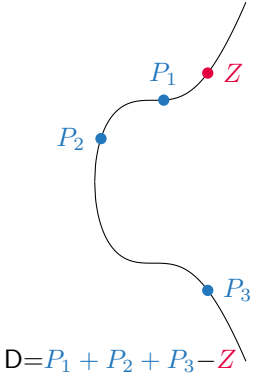
Construction of **good AG codes** relies on $\begin{cases} \textbf{identify algebraic curves} \text{ suitable to the context,} \\ \textbf{design efficient algorithms} \text{ for implementation.} \end{cases}$

TODAY: Computing Riemann–Roch spaces of curves.

[1] La Guardia, Pereira, Quantum Information Processing, 2017
[2] R. Cramer, M. Rambaud and C. Xing, Crypto 2021
[3] S. Bordage, M. Lhotel, J. Nardi and H. Randriam, CCC 2022

Riemann–Roch spaces of curves

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



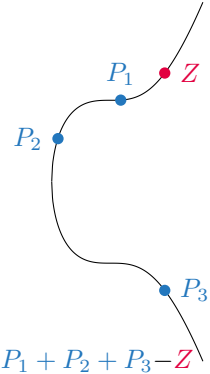
The **Riemann–Roch space** $L(D)$ is the space of functions $\frac{G}{H} \in \mathbb{K}(\mathcal{C})$ such that:

- 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 $\mathcal{C}$: $D = \sum_{P \in \mathcal{C}} n_P P,\ n_P \in \mathbb{Z}$



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

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

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

Toy example

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

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

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

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

$$\rightarrow f \text{ generates the space of solutions.}$$

Toy example

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

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

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

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

$$\rightarrow f \text{ generates the space of solutions.}$$

Strategy

Denominator H vanishes at P: $H(X,Y,1) \equiv 0 \mod X$
Numerator G vanishes at Q: $G(X,Y,1) \equiv 0 \mod X-1$
We retrieve the solution: $\frac{X-1}{X}$.

Brill–Noether method 1874

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*)

Brill–Noether method 1874

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*)

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 1874

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*)

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 1874

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*)

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 1874

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*)

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

Sketch of the algorithm

Input

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

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

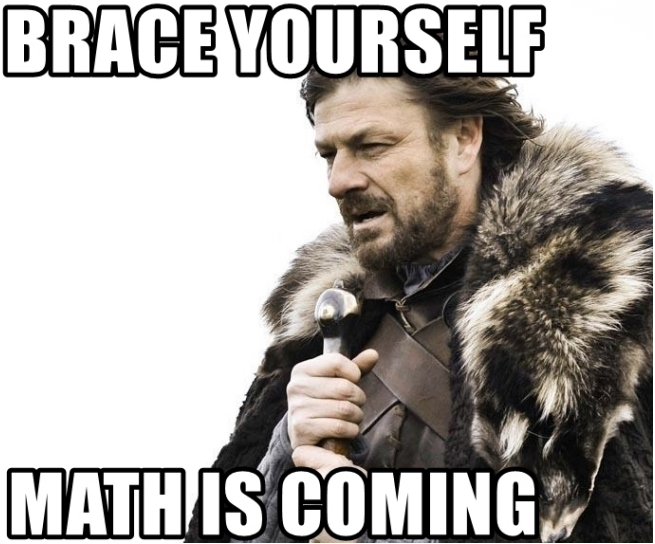
Step 2 : Compute the common denominator H

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.


BRACE YOURSELF
MATH IS COMING

The adjoint divisor

Let $\mathcal{C} : F = 0$ be a plane curve. Let $P \in \mathrm{Sing}(\mathcal{C})$ of multiplicity m, w.l.o.g. $P = (0 : 0 : 1)$.

Definition

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

The adjoint divisor

Let $\mathcal{C} : F = 0$ be a plane curve. Let $P \in \mathrm{Sing}(\mathcal{C})$ of multiplicity m, w.l.o.g. $P = (0:0:1)$.

Definition

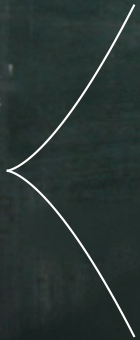
The local adjoint divisor is $\mathcal{A}_P = -\sum_{\mathcal{P}|P} \mathrm{val}_{\mathcal{P}} \left(\frac{dx}{F_y(x,y,1)} \right) \mathcal{P}$.

The polynomial 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, φ_i Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

Example



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

$(0,0)$ unique singular point

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

The adjoint divisor

Let $\mathcal{C} : F = 0$ be a plane curve. Let $P \in \mathrm{Sing}(\mathcal{C})$ of multiplicity m, w.l.o.g. $P = (0 : 0 : 1)$.

Definition

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

The polynomial 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, φ_i Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

The adjoint divisor

Let $\mathcal{C} : F = 0$ be a plane curve. Let $P \in \mathrm{Sing}(\mathcal{C})$ of multiplicity m, w.l.o.g. $P = (0:0:1)$.

Definition

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

The polynomial 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, φ_i Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

Puiseux series		Rational Puiseux Expansions		places $\mathcal{P}$
$\{\varphi_1, \ldots, \varphi_m\}$	$\rightsquigarrow$	$(\gamma t^{e_i}, Y_i(t))$ $i \in \{1, \ldots, s\}$, $s \leqslant m$	$\longleftrightarrow$	

The adjoint divisor

Let $\mathcal{C} : F = 0$ be a plane curve. Let $P \in \mathrm{Sing}(\mathcal{C})$ of multiplicity m, w.l.o.g. $P = (0:0:1)$.

Definition

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

The polynomial 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, φ_i Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

Puiseux series		Rational Puiseux Expansions		places $\mathcal{P}$
$\{\varphi_1, \dots, \varphi_m\}$	$\rightsquigarrow$	$(\gamma t^{e_i}, Y_i(t))$ $i \in \{1, \dots, s\}$, $s \leqslant m$	$\longleftrightarrow$	

The local adjoint divisor becomes

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

Example

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

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

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

Example

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

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

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

$\Longrightarrow (X(t), Y(t)) = (t^2, t^3)$

Example

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

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

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

$\Longrightarrow (X(t), Y(t)) = (t^2, t^3)$

<u>Adjoint condition:</u> $x = t^2 \Rightarrow dx = 2t$, $F_y = 2y$

Example

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

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

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

$\Longrightarrow (X(t), Y(t)) = (t^2, t^3)$

<u>Adjoint condition:</u> $x = t^2 \Rightarrow dx = 2t,\ F_y = 2y$

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

Example

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

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

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

$\Longrightarrow (X(t), Y(t)) = (t^2, t^3)$

<u>Adjoint condition:</u> $x = t^2 \Rightarrow dx = 2t,\ F_y = 2y$

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

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

Sketch of the algorithm

Input

$\mathcal{C} : F(X, Y, Z) = 0$ a plane curve of degree δ, 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 $(H) - D$

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 δ, 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 $(H) - D$

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 $d := \deg H$.

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

[4] $2 \leqslant \omega \leqslant 3$ 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 in the coefficients of H with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

[4] $2 \leqslant \omega \leqslant 3$ is a feasible exponent for linear algebra ($\omega = 2.373$)

Find a denominator in practice: classical linear algebra

Let $d := \deg H$.

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

$\rightsquigarrow$ linear system in the coefficients of H with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

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

$$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega) \text{ operations}^4.$$

[4] $2 \leqslant \omega \leqslant 3$ 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 in the coefficients of H with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

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

$$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega) \text{ operations}^4.$$

How big is d?

[4] $2 \leqslant \omega \leqslant 3$ 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 in the coefficients of H with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

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

$$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega) \text{ operations}^4.$$

How big is d?

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

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

[4] $2 \leqslant \omega \leqslant 3$ is a feasible exponent for linear algebra ($\omega = 2.373$)

Sketch of the algorithm

Input

$\mathcal{C} : F(X, Y, Z) = 0$ *a plane curve of degree* δ, 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 $(H) - D$

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 δ, 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 δ, 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)
$\deg G_i = \deg H$, G_i prime with F and $(G_i) \geqslant (H) - D$

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 δ, 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 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 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 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 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 in positive "small" characteristic (in progress).

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

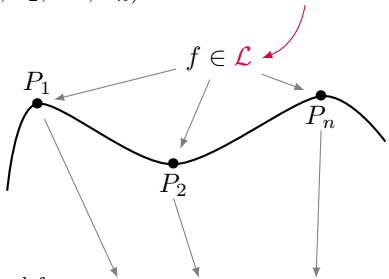
Future questions

- Computing Riemann–Roch spaces in positive "small" characteristic (in progress).

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

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

Riemann–Roch space of the curve



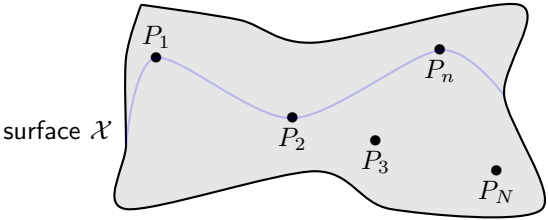
$$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in \mathcal{L}\}$$

Future questions

- Computing Riemann–Roch spaces in positive "small" characteristic (in progress).

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

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

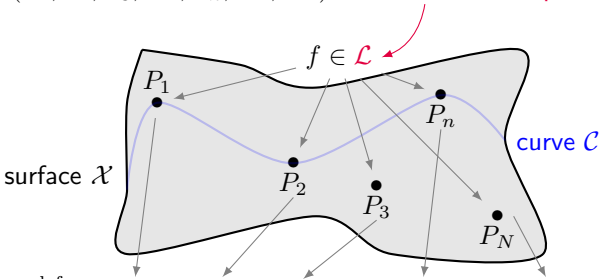


Future questions

- Computing Riemann–Roch spaces in positive "small" characteristic (in progress).

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

$\mathcal{P} = (P_1, P_2, P_3, \ldots, P_n, \ldots, P_N)$ 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)
- ✔ Local properties from curves on $\mathcal{X}$
  (*e.g.* local recoverability)

Future questions

- Computing Riemann–Roch spaces in positive "small" characteristic (in progress).

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

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

Future questions

- Computing Riemann–Roch spaces in positive "small" characteristic (in progress).

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

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

Thank you for your attention!

Questions? e.berardini@tue.nl