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"

Coding theory and cryptography
A conference in honor of Joachim Rosenthal's 60th birthday
12 July 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**

$$\left.\begin{array}{c}\text{dimension} \leftrightarrow \text{information} \\ \text{minimum distance} \leftrightarrow \text{correction capacity}\end{array}\right\} \quad k + d \leqslant n + 1$$ Singleton, 1964

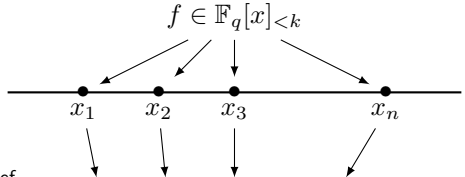
Linear codes: from Reed–Solomon codes...

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**

$$\left.\begin{array}{c}\text{dimension} \leftrightarrow \text{information} \\ \text{minimum distance} \leftrightarrow \text{correction capacity}\end{array}\right\} \quad k + d \leqslant n + 1$$ Singleton, 1964

Reed–Solomon (RS) Codes Reed and Solomon, 1960



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

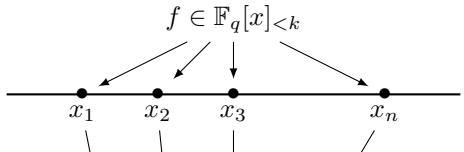
Linear codes: from Reed–Solomon codes...

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



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

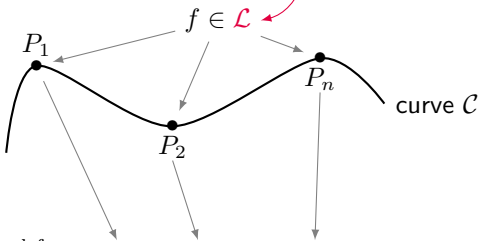
- ✔ **Optimal parameters**
  $k + d = n + 1$.
- ✔ **Effective decoding algorithms**
  Berlekamp,1968.
- ⚠ **Drawback:** $n \leqslant q$.

  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)

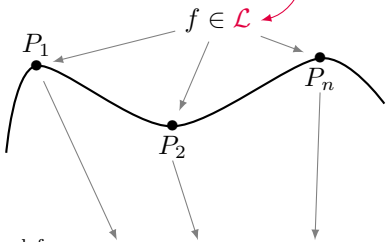


$C_{\mathcal{C}}(\mathcal{L}, \mathcal{P}) \stackrel{\text{def}}{=} \{(f(P_1), f(P_2), \ldots, f(P_n)) \mid f \in L\}$

...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** $\mathcal{C}$

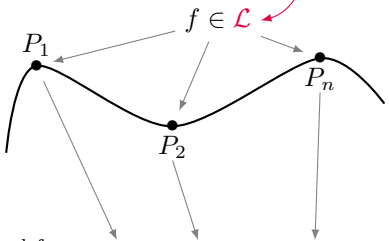
- ✔ **Good parameters**
- ✔ **Efficient decoding algorithms**
- ✔ **Length** $> q$

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

...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** $> q$

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

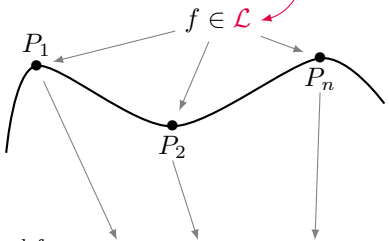
Proposition

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]
- 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]

[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]
- 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]

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]
- 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]

Construction of **good AG codes** relies on
- **identify algebraic curves** suitable to the context,
- **design efficient algorithms** for implementation.

[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]
- 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]

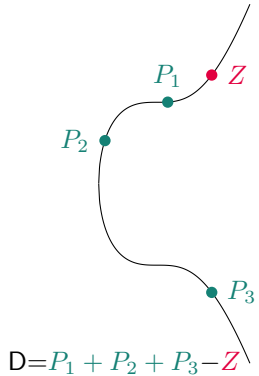
Construction of **good AG codes** relies on $\begin{cases} \textbf{identify algebraic curves} \text{ suitable to the context,} \\ \textbf{design efficient algorithms} \text{ for implementation.} \end{cases}$

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



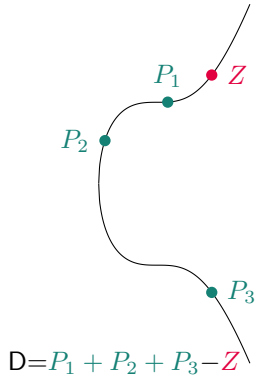
The **Riemann–Roch space** $L(D)$ is the space of functions $\frac{G}{H} \in \mathbb{K}(\mathcal{C})$ such that:

- 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

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

$f = \frac{X-1}{X}$ 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.

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

$\rightarrow f$ 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

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

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

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

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

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

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:

- $(H) = \sum_{P \in \mathcal{C}} \operatorname{ord}_P(H) P$ – 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:

- $(H) = \sum_{P \in \mathcal{C}} \operatorname{ord}_P(H) P$ – 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:

- $(H) = \sum_{P \in \mathcal{C}} \operatorname{ord}_P(H) P$ – 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:

- $(H) = \sum_{P \in \mathcal{C}} \operatorname{ord}_P(H) P$ – 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

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 $\mathcal{A}$

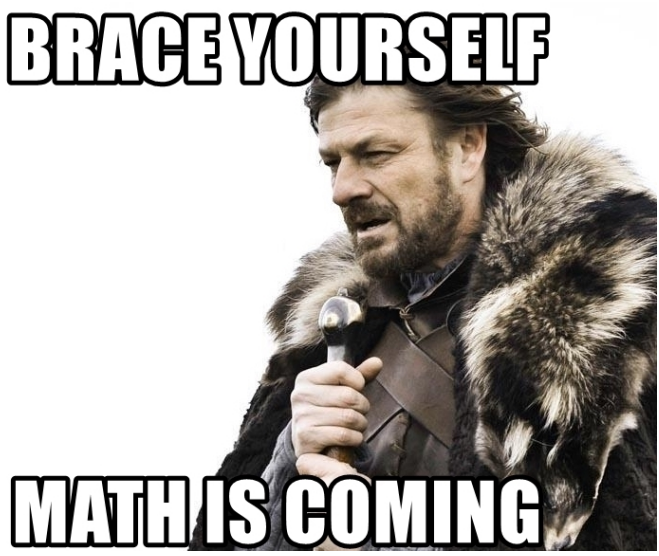
Step 2 : Compute the common denominator H

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

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, 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 $u \in \overline{\mathbb{K}}[[x,y]]$ 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, 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 $u \in \overline{\mathbb{K}}[[x,y]]$ 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 of $\mathcal{C}$

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, 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 $u \in \overline{\mathbb{K}}[[x,y]]$ 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 of $\mathcal{C}$

The *local adjoint divisor* becomes

$$\mathcal{A}_P = (m-1)\sum_{i=1}^{m}\mathcal{P}_i.$$

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 φ of degree e, ζ 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 φ of degree e, ζ 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 φ of degree e, ζ 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 $\mathcal{C}$ 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 φ_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 φ_i Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

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

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

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

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 $P \in \mathrm{Sing}(\mathcal{C})$ ~~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 φ_i Puiseux series of $F \in \overline{\mathbb{K}}[[x]][y]$.

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

The local adjoint divisor becomes

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

In practice: algorithm for computing Puiseux series[4] $\rightsquigarrow$ $\mathcal{A}$ computed with $\tilde{O}(\delta^3)$ operations.

[4] A. Poteaux and M. Weimann, Annales Henri Lebesgue, 2021

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 $d := \deg H$.

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

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

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

$$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega) \text{ operations}^5 \text{ in } \mathbb{K}.$$

[5] $2 \leqslant \omega \leqslant 3$ 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 with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

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

$$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega) \text{ operations}^5 \text{ in } \mathbb{K}.$$

How big is d?

[5] $2 \leqslant \omega \leqslant 3$ 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 with $\deg \mathcal{A} + \deg D_+ \sim \delta^2 + \deg D_+$ equations,

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

$$\tilde{O}((d\delta + \delta^2 + \deg D)^\omega) \text{ operations}^5 \text{ in } \mathbb{K}.$$

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 $\mathbb{K}$.

[5] $2 \leqslant \omega \leqslant 3$ is a feasible exponent for linear algebra ($\omega = 2.373$)

Second method: structured linear algebra

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

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

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

[6] 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[6] 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.

[6] 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* 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)

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* $\tilde{\mathcal{O}}((\delta^2 + \deg D_+)^\omega)$ *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 *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 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.

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

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.

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

Thank you for your attention!

Questions? e.berardini@tue.nl