Riemann–Roch spaces & Algebraic Geometry codes

Elena Berardini

Télécom Paris, Institut Polytechnique de Paris, France

Cryptography and Coding Theory
First Annual Conference
21st September 2021

What is an error correcting code?

A tool for transmitting and storing data.

Main feature: detection and correction of the errors that can occur during transmission/storage

What is an error correcting code?

A tool for transmitting and storing data.

Main feature: detection and correction of the errors that can occur during transmission/storage

A $\mathbb{F}_q$-vector subspace of $\mathbb{F}_q^n$ (linear codes).

Three parameters:

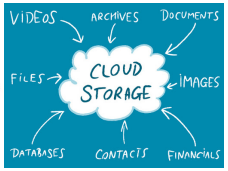
- **n**, the length
- **k**, the dimension
- **d**, the minimum distance

Rate of transmission: k/n
Detects up to $d-1$ errors
Corrects up to $\lfloor \frac{d-1}{2} \rfloor$ errors

What is an error correcting code?

A tool for transmitting and storing data.

Main feature: detection and correction of the errors that can occur during transmission/storage



A $\mathbb{F}_q$-vector subspace of $\mathbb{F}_q^n$ (linear codes).

Three parameters:

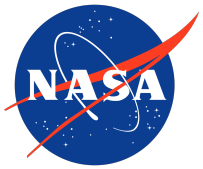
- **n**, the length
- **k**, the dimension
- **d**, the minimum distance

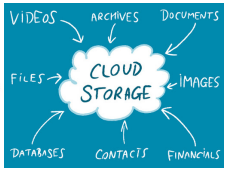
Rate of transmission: k/n
Detects up to $d-1$ errors
Corrects up to $\lfloor \frac{d-1}{2} \rfloor$ errors

What is an error correcting code?

A tool for transmitting and storing data.

Main feature: detection and correction of the errors that can occur during transmission/storage





GOAL: to encode as much data as possible and to detect and correct as many errors as possible!

A $\mathbb{F}_q$-vector subspace of $\mathbb{F}_q^n$ (linear codes).

Three parameters:

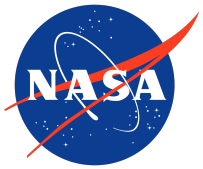
- **n**, the length
- **k**, the dimension
- **d**, the minimum distance

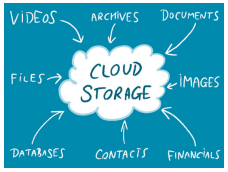
Rate of transmission: k/n
Detects up to $d-1$ errors
Corrects up to $\lfloor \frac{d-1}{2} \rfloor$ errors

What is an error correcting code?

A tool for transmitting and storing data.

Main feature: detection and correction of the errors that can occur during transmission/storage





GOAL: to encode as much data as possible and to detect and correct as many errors as possible!

A $\mathbb{F}_q$-vector subspace of $\mathbb{F}_q^n$ (linear codes).

Three parameters:

- **n**, the length
- **k**, the dimension
- **d**, the minimum distance

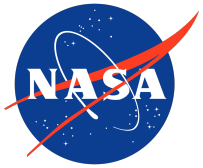
Rate of transmission: k/n
Detects up to $d-1$ errors
Corrects up to $\lfloor \frac{d-1}{2} \rfloor$ errors

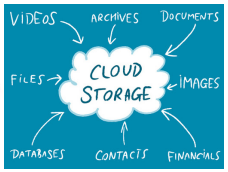
GOAL: to have **k** and **d** as big as possible!

What is an error correcting code?

A tool for transmitting and storing data.

Main feature: detection and correction of the errors that can occur during transmission/storage





GOAL: to encode as much data as possible and to detect and correct as many errors as possible!

A $\mathbb{F}_q$-vector subspace of $\mathbb{F}_q^n$ (linear codes).

Three parameters:

- **n**, the length
- **k**, the dimension
- **d**, the minimum distance

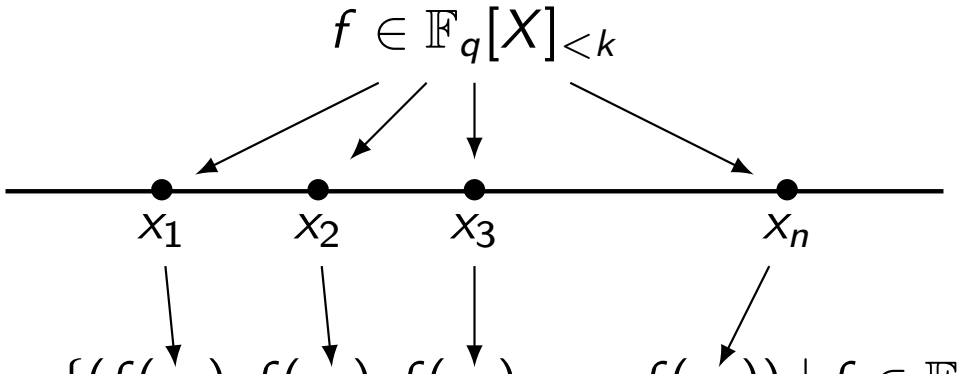
Rate of transmission: k/n
Detects up to $d-1$ errors
Corrects up to $\lfloor \frac{d-1}{2} \rfloor$ errors

GOAL: to have **k** and **d** as big as possible!

Singleton Bound: $k + d \leqslant n + 1$
$\rightsquigarrow$ tradeoff between redundancy and capacity of errors-correction

Evaluation codes: from Reed–Solomon to AG codes

Reed-Solomon codes:



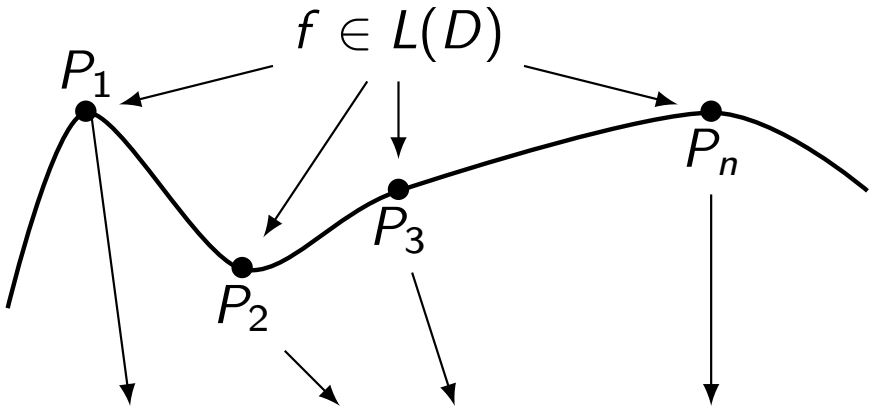
$$\mathrm{RS}_k(\mathbf{x}) := \{(f(x_1), f(x_2), f(x_3), \ldots, f(x_n)) \mid f \in \mathbb{F}_q[X]_{<k}\}$$

✓ Optimal parameters: $k + d = n + 1$ (MDS codes)

⚠ Drawback: require $n < q$

Evaluation codes: from Reed–Solomon to AG codes

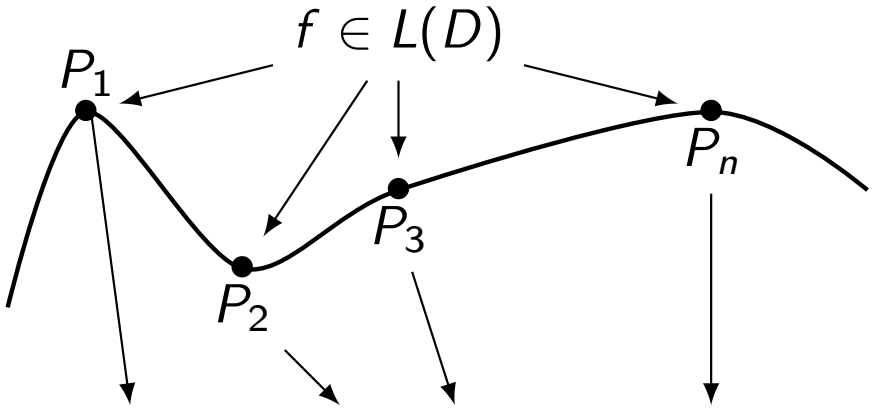
$\rightsquigarrow$ **Algebraic geometry codes:**



$$\mathcal{C}((P_i)_i, D) := \{(f(P_1), f(P_2), f(P_3), \ldots, f(P_n)) \mid f \in L(D)\}$$

Evaluation codes: from Reed–Solomon to AG codes

$\rightsquigarrow$ **Algebraic geometry codes:**



$$\mathcal{C}((P_i)_i, D) := \{(f(P_1), f(P_2), f(P_3), \ldots, f(P_n)) \mid f \in L(D)\}$$

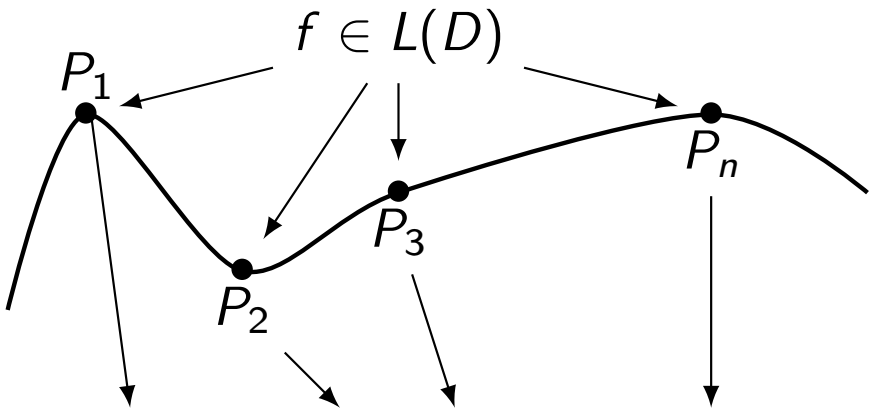
(Some) Recent applications of AG codes:

- Locally Recoverable Codes[1]
- Interactive Oracle Proofs[2]

[1] A. Barg, I. Tamo and S. Vladuts, IEEE Transactions on Information Theory, 2017
[2] S. Bordage and J. Nardi, preprint, 2020

Evaluation codes: from Reed–Solomon to AG codes

$\rightsquigarrow$ **Algebraic geometry codes:**



$$\mathcal{C}((P_i)_i, D) := \{(f(P_1), f(P_2), f(P_3), \ldots, f(P_n)) \mid f \in L(D)\}$$

(Some) Recent applications of AG codes:

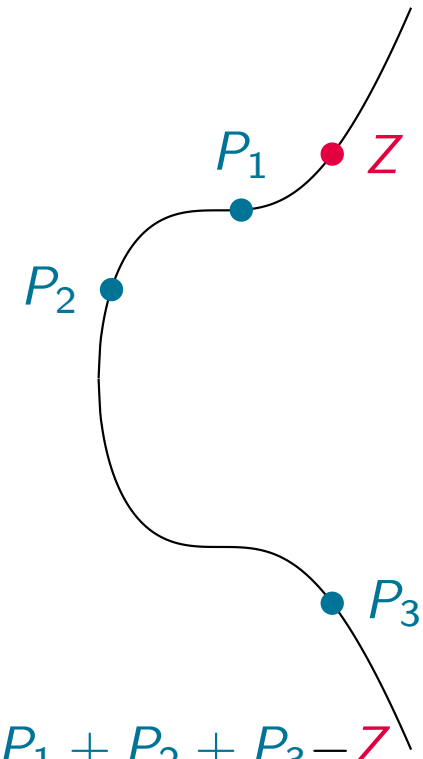
- Locally Recoverable Codes[1]
- Interactive Oracle Proofs[2]

Explicit construction of AG codes $\rightsquigarrow$ need of explicit computation of $L(D)$

[1] A. Barg, I. Tamo and S. Vladuts, IEEE Transactions on Information Theory, 2017
[2] S. Bordage and J. Nardi, preprint, 2020

Riemann–Roch space

Divisor on a curve $\mathcal{C}$: $D = \sum_{P \in \mathcal{C}} n_P P$



D=$P_1 + P_2 + P_3 - Z$

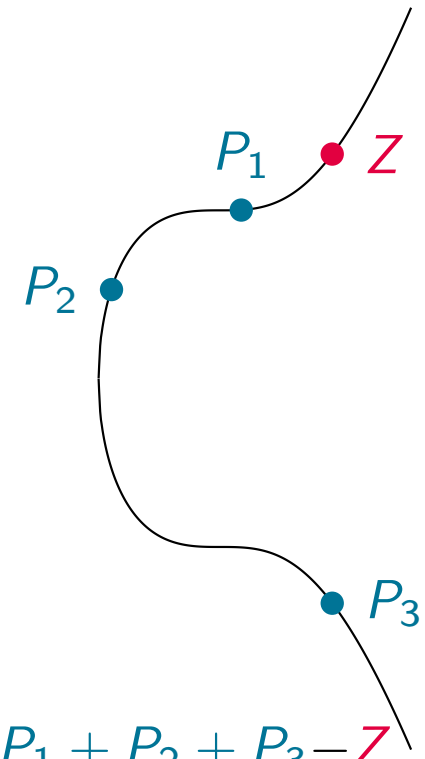
The **Riemann–Roch space** $L(D)$ is the space of all functions $\frac{G}{H} \in \mathbb{K}(\mathcal{C})$ s. t.:

- 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 not other poles outside the points P with $n_P > 0$

Here: Z must be a zero of G, the P_i's can be zeros of H

Riemann–Roch space

Divisor on a curve $\mathcal{C}$: $D = \sum_{P \in \mathcal{C}} n_P P$



D=$P_1 + P_2 + P_3 - Z$

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

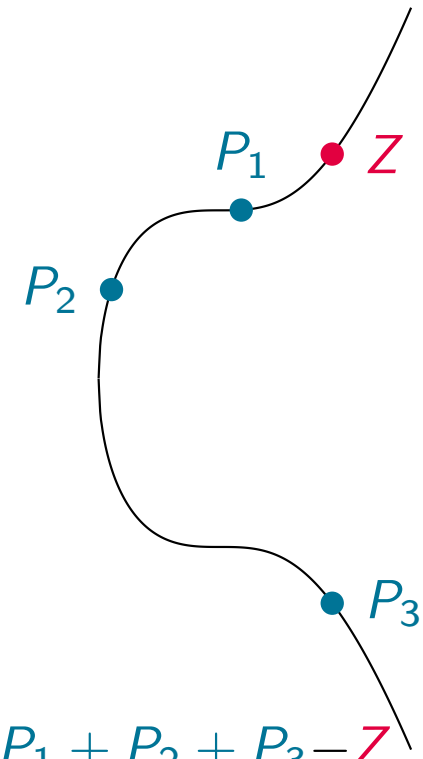
- if $n_P < 0$ then P must be a zero of G (of multiplicity $\geqslant -n_P$)
- if $n_P > 0$ then P can be a zero of H (of multiplicity $\leqslant n_P$)
- G/H has not other poles outside the points P with $n_P > 0$

Here: Z must be a zero of G, the P_i's can be zeros of H

Riemann–Roch theorem $\rightsquigarrow$ dimension of $L(D)$

Riemann–Roch space

Divisor on a curve $\mathcal{C}$: $D = \sum_{P \in \mathcal{C}} n_P P$



D=$P_1 + P_2 + P_3 - Z$

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

- 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 not other poles outside the points P with $n_P > 0$

Here: Z must be a zero of G, the P_i's can be zeros of H

Riemann–Roch theorem $\rightsquigarrow$ dimension of $L(D)$

⚠no explicit method to compute a basis of $L(D)$

Riemann-Roch problem: state of the art

Geometric methods:
(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 methods:
(Ideals in function fields)
- Hensel–Landberg (1902)
- Coates (1970)
- Davenport (1981)
- Hess (2001)

Riemann-Roch problem: state of the art

Geometric methods:
(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 methods:
(Ideals in function fields)
- Hensel–Landberg (1902)
- Coates (1970)
- Davenport (1981)
- Hess (2001)

Nodal/ordinary curves:	Las Vegas algorithm computing $L(D)$ in $\tilde{\mathcal{O}}((\delta^2 + \deg D_+)^{\frac{\omega+1}{2}})$ field operations[3]
Non-ordinary curves:	⚠no explicit complexity exponent

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

Brill–Noether in a nutshell

Brill–Noether method $\rightsquigarrow$ NSC on H and G such that $G/H \in L(D)$

Let $\mathcal{C} : F(X, Y, Z) = 0$ be a plane projective curve and D a smooth divisor on it.

<u>Notation:</u> (H) = zeros of H with multiplicity

Description of $L(D)$: non-zero elements are of the form $\frac{G_i}{H}$ where

- H satisfies $(H) \geqslant D$
- H passes through all the singular points of $\mathcal{C}$ with ad hoc multiplicities
- $\deg G_i = \deg H$, G_i coprime with F and $(G_i) \geqslant (H) - D$

Brill–Noether in a nutshell

Brill–Noether method $\rightsquigarrow$ NSC on H and G such that $G/H \in L(D)$

Let $\mathcal{C} : F(X, Y, Z) = 0$ be a plane projective curve and D a smooth divisor on it.

Notation: (H) = zeros of H with multiplicity

Description of $L(D)$: non-zero elements are of the form $\frac{G_i}{H}$ where

- H satisfies $(H) \geqslant D$
- H passes through all the singular points of $\mathcal{C}$ with ad hoc multiplicities
- $\deg G_i = \deg H$, G_i coprime with F and $(G_i) \geqslant (H) - D$

How do we handle singular points?

Brill–Noether in a nutshell

Brill–Noether method $\rightsquigarrow$ NSC on H and G such that $G/H \in L(D)$

Let $\mathcal{C} : F(X, Y, Z) = 0$ be a plane projective curve and D a smooth divisor on it.

Notation: (H) = zeros of H with multiplicity

Description of $L(D)$: 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 coprime with F and $(G_i) \geqslant (H) - D$

How do we handle singular points?

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

Brill–Noether in a nutshell

Brill–Noether method $\rightsquigarrow$ NSC on H and G such that $G/H \in L(D)$

Let $\mathcal{C} : F(X, Y, Z) = 0$ be a plane projective curve and D a smooth divisor on it.

Notation: (H) = zeros of H with multiplicity

Description of $L(D)$: 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 coprime with F and $(G_i) \geqslant (H) - D$

How do we handle singular points?

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

How do we handle divisors?

Brill–Noether in a nutshell

Brill–Noether method $\rightsquigarrow$ NSC on H and G such that $G/H \in L(D)$

Let $\mathcal{C} : F(X, Y, Z) = 0$ be a plane projective curve and D a smooth divisor on it.

Notation: (H) = zeros of H with multiplicity

Description of $L(D)$: 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 coprime with F and $(G_i) \geqslant (H) - D$

How do we handle singular points?

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

How do we handle divisors?

series expansions of multi-set representations $((P_i)_i, m_i)$	$\rightsquigarrow$	routines on divisors have negligible cost

Sketch of the algorithm

Input

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

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

Step 2: Compute a common denominator H

Step 3: Compute $(H) - D$

Step 4: Compute numerators G_i (similar to Step 2)

Output

A basis of the Riemann–Roch space $L(D)$.

Non-ordinary curves: an explicit complexity exponent

Adjoint divisor: representation in terms of the Puiseux expansions $(X(t), Y(t))$ in the neighborhoods of the singular points

$\rightsquigarrow$ fast algorithms for Puiseux series expansions of germs of curves[4]

Conditions $(H) \geqslant \mathcal{A} + D$ and $(G_i) \geqslant (H) - D$:

$\rightsquigarrow$ linear system

or

$\mathrm{val}_t(H(X(t), Y(X))$ sufficiently large

$\rightsquigarrow$ $K[t]$-module structure

$\rightsquigarrow$ structured linear algebra algorithm[5]

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

[5] C.-P. Jeannerod, V. Neiger, E. Schost and G. Villard, Journal of Symbolic Computation, 2017

Non-ordinary curves: an explicit complexity exponent

Adjoint divisor: representation in terms of the Puiseux expansions $(X(t), Y(t))$ in the neighborhoods of the singular points

$\rightsquigarrow$ fast algorithms for Puiseux series expansions of germs of curves[4]

Conditions $(H) \geqslant \mathcal{A} + D$ and $(G_i) \geqslant (H) - D$:

$\rightsquigarrow$ linear system

or

$\mathrm{val}_t(H(X(t), Y(X))$ sufficiently large

$\rightsquigarrow$ $K[t]$-module structure

$\rightsquigarrow$ structured linear algebra algorithm[5]

Theorem (Abelard, B., Couvreur, Lecerf [6])

Las Vegas algorithm computing $L(D)$ in $\tilde{\mathcal{O}}((\delta^2 + \deg D_+)^\omega)$ field operations.

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

[5] C.-P. Jeannerod, V. Neiger, E. Schost and G. Villard, Journal of Symbolic Computation, 2017

[6] S. Abelard, E. Berardini, A. Couvreur and G. Lecerf, preprint 2021

From curves to surfaces

	Curves	Surfaces
Divisors	sum of points	sum of curves

Riemann-Roch spaces of surfaces are again spaces of functions
$\rightsquigarrow$ same construction of codes from curves holds for codes from surfaces!

From curves to surfaces

	Curves	Surfaces
Divisors	sum of points	sum of curves

Riemann-Roch spaces of surfaces are again spaces of functions
$\rightsquigarrow$ same construction of codes from curves holds for codes from surfaces!

Why codes from surfaces?

Number of rational points: $O(q^2)$ / surface VS $O(q)$ / curve
$\rightsquigarrow$ construction of codes of same length on smaller finite fields

Applications: codes from surfaces provided optimal Local Recoverable Codes [7] (for Distributed Storage Systems)

Mathematical interest: new mathematical questions arise from the study of AG codes from surfaces

[7] C. Salgado, A. Várilly-Alvarado and J. F. Voloch, preprint 2019

Study of AG codes from surfaces

The length of the codes

$$n = \text{number of rational points on the surface}$$

The dimension of the codes

Riemann–Roch theorem for surfaces $\rightsquigarrow$ dimension of the code
⚠still does not give an effective method to compute a basis of $L(D)$!

The minimum distance of the codes

We can prove that

$$d \geqslant n - \max_{f \in L(D) \setminus \{0\}} \sum_{i=1}^{k} \#\mathcal{C}_i(\mathbb{F}_q)$$

where the $\mathcal{C}_i$ are irreducible curves on the surface.

Algebraic geometry tools enter the game

Bounding k:

- Can be done using intersection theory on surfaces.

Bounding $\#\mathcal{C}(\mathbb{F}_q)$:

- Different bounds already exist and can be used in this context.
- More precise upper bounds for $\#\mathcal{C}(\mathbb{F}_q)$ for curves on surfaces will lead to more precise lower bounds for the minimum distance.

The bound on the minimum distance
depends on invariant of the surface

$$\Downarrow$$

**We get hints on which surfaces
are more suitable for AG codes**

Examples of results on the minimum distance

- **Abelian surfaces**[8] without irreducible curves of genus $\pi \leqslant \ell$:

$$d(\mathcal{X}, D) \geqslant n - \sqrt{\frac{D^2}{2\ell}}\,(q + 1 - \mathrm{Tr}(\mathcal{X}) + (\ell - 1)\lfloor 2\sqrt{q} \rfloor)$$

$\rightsquigarrow$ **better bound for big ℓ!**

- **Fibered surfaces**[9] on a base curve B:

$$d(\mathcal{X}, D) \geqslant d^*(\mathcal{X}, D) + \delta(B),$$

where $\delta(B) := q + 1 + g_B \lfloor 2\sqrt{q} \rfloor - \#B(\mathbb{F}_q) \geqslant 0$.

$\rightsquigarrow$ **better bound if B has few rational points!**

[8] Y. Aubry, E. Berardini, F. Herbaut and M. Perret, Finite Fields Appl. 70, 2021
[9] Y. Aubry, E. Berardini, F. Herbaut and M. Perret, Contemp. Maths. 770, 2021

What's next?

AG codes from curves.

- Implementation including fast structured linear algebra.
- Computing Riemann–Roch spaces of non-ordinary curves in "small" positive characteristic (in progress with A. Couvreur and G. Lecerf)
- Improving the complexity in the non-ordinary case ($\rightsquigarrow$ sub-quadratic?)

AG codes in higher dimension.

- Use algebraic geometry methods to study codes from 3-folds.
- Compute Riemann–Roch spaces of surfaces $\rightsquigarrow$ explicit construction of (good) AG codes from surfaces.

GV2DB ODJ XDF ODJF 2GG4DGTDD!*

Questions?

elena.berardini@telecom-paris.fr

*Thank you for your attention!