A Geometric Approach for the Computation of Riemann-Roch Spaces : Algorithm and Complexity

Aude Le Gluher and Pierre-Jean Spaenlehauer
Université de Lorraine / INRIA Nancy – Grand Est / CNRS
CARAMBA team

SPECFUN team, 2019

Setup : the Riemman-Roch Problem

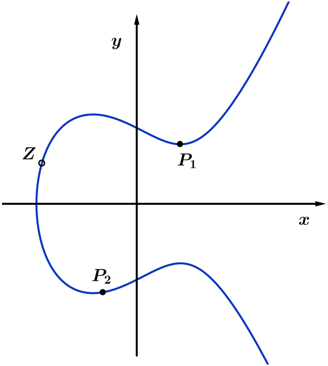
$\mathbf{K}$: sufficiently large perfect field.

C : irreducible projective nodal curve with r nodes, described by $Q \in \mathbf{K}[X, Y]$.

Setup : the Riemman-Roch Problem

$\mathbf{K}$: sufficiently large perfect field.

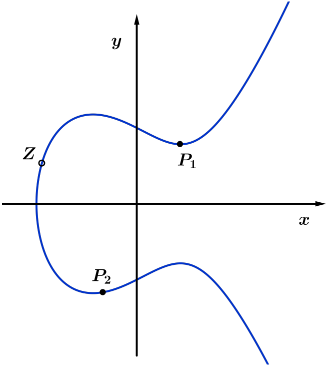
C : irreducible projective nodal curve with r nodes, described by $Q \in \mathbf{K}[X, Y]$.



Setup : the Riemman-Roch Problem

K : sufficiently large perfect field.

C : irreducible projective nodal curve with r nodes, described by $Q \in \mathbf{K}[X, Y]$.



Goal : find all functions $R(X,Y)/S(X,Y) \in \mathbf{K}(C) = \mathrm{Frac}(\mathbf{K}[X,Y]/(Q))$ such that :

$$\begin{cases} R(Z) = 0 \\ S \text{ may cancel at } P_1 \\ S \text{ may cancel at } P_2 \\ \text{ no poles at infinity} \end{cases}$$

$\boxed{\text{Prescribed zeroes, authorized poles}}$

Motivations

Riemann-Roch spaces are vector spaces useful in particular for :

- Computing the group law of the Jacobian of a curve.
  Volcheck (1994), Huang et Ierardi (1994), Khuri-Makdisi (1995).
- Building algebraic geometric error-correcting codes.
  Goppa (1983), Haché (1995).
- Integration of algebraic functions. Davenport (1981).

State of the art

Here, C is a curve of degree d and genus g and $D = D_+ - D_-$ is a divisor on C.

Computation of general Riemann-Roch spaces :

- Huang and Ierardi (1994) : geometric algorithm in $O(d^6 \deg(D_+)^6)$.
- Haché (1995).
- Hess's arithmetic algorithm (2002).

Computation of the group law in Jacobians ($\deg(D_+) = O(g)$) :

- Volcheck (1994) : arithmetic algorithm in $O(\max(d, g)^7)$.
- Khuri-Makdisi (2007) : algorithm in $O(g^{\omega+\varepsilon})$ where ω is a feasible exponent for matrix multiplication and $\varepsilon > 0$.
- Possible improvements for specific curves (for instance $\widetilde{O}(g)$ for hyperelliptic curves, Cantor).

Main results

- Variant of the Brill-Noether algorithm : geometric probabilistic algorithm for computing Riemann-Roch spaces in the case of divisors not involving singular points. Mild assumption when the curve is singular.
- Bound on the probability of failure :

  $$O(\max(\deg(C)^4, \deg(D_+)^2)/|E|)$$

  where E is a finite subset of $\mathbf{K}$ in which we can pick elements at random uniformly.
- Proof of complexity :
  Number of arithmetic operations in $\mathbf{K}$ bounded by :

  $$O(\max(\deg(C)^{2\omega}, \deg(D_+)^{\omega})$$

  where ω is a feasible exponent for matrix multiplication.
- C++/NTL implementation of this algorithm.

Plan

Divisors and the Riemann-Roch problem

- A divisor D on C is a formal sum with integer coefficients of closed points.
- The divisor associated to a function $g \in \mathbf{K}(C)$ is the divisor for which the coefficient of $P \in C$ is the valuation of g in P.
- The nodal divisor E is the divisor of degree $2r$ which is the sum of the points that project to a node.
- The Riemann-Roch space associated to the divisor D is

$$L(D) = \{f \in \mathbf{K}(C) \setminus \{0\} | (f) \geq -D\} \cup \{0\}$$

Keep in mind

When writing $D = D_+ - D_-$, the divisor D_+ constrains the poles of $f \in L(D)$ and D_- constrains its zeroes.

Input and output

Input :

- A polynomial $q \in \mathbf{K}[X, Y]$ describing an irreducible projective plane curve C.
- The representation of the nodal divisor E.
- The representations of two effective smooth divisors D_+ and D_-.

Output : A basis of the vector space $L(D)$ where $D = D_+ - D_-$.

(Mild) assumptions on the input

- The polynomial q is monic in Y.
- The degree in Y of q equals its total degree.

Mild assumptions because...

This can be enforced by a linear change of coordinates.

- No singular point in the input divisor $D = D_+ - D_-$.
- There exists a form h of a chosen degree d such that $(h) \geq D_+ + E$ and $(h) - E$ does not involve any singular point.

Goal of these assumptions

We want singularities to have a minimal impact on computations.

Construction of a suitable denominator

Common denominator of degree d.

$\longrightarrow$ Choose a random polynomial h of degree d which vanishes with the right multiplicities at all points prescribed by $D_+ + E$: h is solution of an underdetermined linear system.

$\longrightarrow$ Computation of a representation for the effective divisor $(h) - E$.

About the degree of h

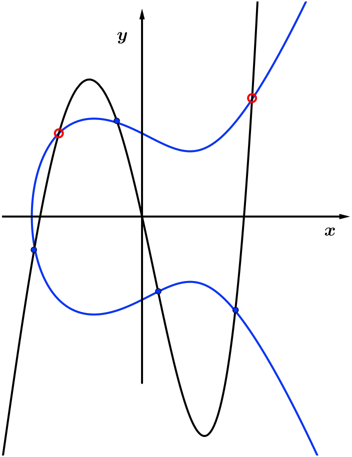
The degree d is tuned to be as small as possible while guaranteeing an underdetermined linear system. We have :

$$d < \frac{\deg(D_+) + r}{\deg(C)} + \deg(C)$$

Why bother with E ?

We need h to vanish at singularities to use Brill-Noether residue theorem.

Readjusting the zeroes



Non exact interpolation : h has non desired smooth zeroes.

$\longrightarrow$ Find those non desired zeroes : they are represented by $[(h) - E] - D_+$.
$\longrightarrow$ Add them to D_-.

Counterbalance the unwanted zeros of the denominator by the same zeros for the numerators.

Importance of our singular assumption

We assume $(h) - E$ does not involve any singular point of C.

Construction of the numerators

From last step : $D' = D_- + [(h) - E] - D_+ + E$ imposes the zeros of numerators.

$\longrightarrow$ Computation of a base B of polynomials of degree at most $\deg(h)$ and vanishing at all points prescribed by D' with the right multiplicities : again a linear system.

Correction of the algorithm

The set $\{b/h \mid b \in B\}$ is a base of the Riemann-Roch space $L(D)$.

Proof : $\mathrm{Vect}(\{b/h \mid b \in B\}) \subset L(D)$ by construction. The converse uses the Brill-Noether residue theorem.

Sum up of the algorithm

- Choose an interpolating polynomial h as denominator.
- Compute the representation of the smooth part of (h).
- Identify the unwanted zeros of h.
- Find the new constraints on the zeroes of numerators.
- Compute a base of numerators.

Plan

Main idea

What do we represent ?

We represent effective divisors D with no singular points.

The representation of D is :

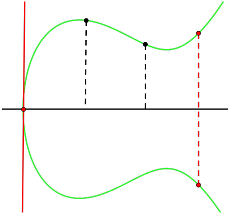
- Similar to Mumford Coordinates in the case of hyperelliptic curves,
- Encodes the effective divisor by univariate polynomials (Giusti, Lecerf, Salvy, 1999). In particular :
- Finds a univariate polynomial χ such that $\mathbf{K}[C]/(I) \cong \mathbf{K}[S]/\chi(S)$ where I is an ideal such that $\mathbf{K}[C]/(I)$ is the description of the algebraic set corresponding to the support of D.

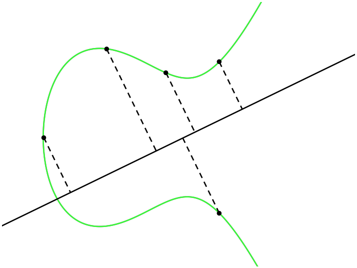
Primitive representation of effective divisors

An effective divisor D is represented by $(\lambda, \chi, u, v) \in \mathbf{K} \times \mathbf{K}[S]^3$ such that :

1. The degree of χ is the degree of D and $\deg(u), \deg(v) < \deg(D)$.
2. $q(u(S), v(S)) \equiv 0 \bmod \chi(S)$.
3. $\lambda u(S) + v(S) = S$.
4. $GCD\left(\frac{\partial q}{\partial X}(u(S), v(S)) - \lambda \frac{\partial q}{\partial Y}(u(S), v(S)), \chi(S)\right) = 1.$

Illustration of the representation





Potential problems :

- Points of the divisor with the same projection.
- Tangents to the curve perpendicular to the direction of projection at some divisor points.

Solution : Find a suitable direction of projection.

Existence of the representation

Warning

Such a representation does not always exist !

BUT

It does exist if the field $\mathbf{K}$ is large enough.

Idea of the proof :

- The $\lambda \in \mathbf{K}$ such that $\lambda X + Y$ is not a primitive element of $\mathbf{K}[C]/(m)$ where m is a maximal ideal representing a point P of C are finite.
- Build representations for each point P involved in the divisor by finding primitive elements of the form $\lambda X + Y$ for $\mathbf{K}[C]/(m)$.
- Lift those representations thanks to Hensel's lemma to encode multiplicities.
- Use the CRT to find the final representation.

Operations needed on smooth representations

Our algorithm requires us to know how to :

- Sum two representations.
- Subtract two representations (knowing that the result will remain an effective divisor).
- Compute the representation of the divisor $(h) - E$.

Remark

The first two operations require the two input representations to agree on a common λ. Need to change the primitive element (Giusti, Lecerf, Salvy, 1999).

Agreeing on a primitive element in practice

In practice, if $\mathbf{K}$ is large enough, a random choice of λ should work :

$\longrightarrow$ By default, choose $\lambda = 0$.

$\longrightarrow$ If an error occurs at some point (bad λ), choose another λ and restart the computations from the top instead of changing the primitive element as we go.

Example : the subtraction

Input : Two representations $(\lambda, \chi_1, u_1, v_1)$ and $(\lambda, \chi_2, u_2, v_2)$ of effective smooth divisors D_1 and D_2.

Output : The representation of $D_1 - D_2$ if this divisor remains effective.

Algorithm :

- Suppress the common factors of χ_1 and χ_2 by computing $\chi = \chi_1 / GCD(\chi_1, \chi_2)$
- Reduce u_1 and v_1 modulo χ.
- Return (λ, χ, u, v).

Main idea

With this representation, operations on divisors are operations on univariate polynomials.

When does the algorithm fails ?

Failure = bad choice for the λ used to represent divisors.

Bound on the probability of failure

Assuming we can choose elements of $\mathbf{K}$ uniformly at random in a finite subset $E \subset \mathbf{K}$, the probability that our algorithm fails is bounded above by

$$O(\max(\deg(C)^4, \deg(D_+)^2)/|E|)$$

Idea of the proof : The set of bad λ is included in the set of roots of a finite number of polynomials. Bounding their degrees concludes.

Plan

Translation of the operations needed

- Choose polynomial h as denominator : build + solve linear system.
- Compute the representation of $(h) - E$: resultant and subresultant.
- Identify the unwanted zeros of h : GCD.
- Find the new constraints on the zeroes of numerators : CRT.
- Compute a base of numerators : build + solve linear system.

Costs of each operation

All complexity bounds count the number of arithmetic operations in $\mathbf{K}$.

- Build + find a solution to the first linear system : $O((\deg(D_+) + r)^\omega)$.
- Resultant and subresultant : $\widetilde{O}(\max(\deg(C)^3, (\deg(D_+) + r)^2/\deg(C)))$.
- GCD's and CRT : both in $O(\max(\deg(C)^{2\omega}, \deg(D_+)^\omega))$.
- Build + solve the second linear system : $O(\max(\deg(C)^{2\omega}, \deg(D_+)^\omega))$.

Linear algebra rules

Both in theory and practice.

Final complexity and comparisons

Final complexity

Our algorithm requires at most

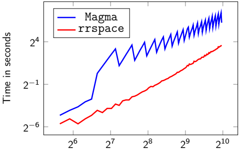
$$O(\max(\deg(C)^{2\omega}, \deg(D_+)^{\omega})$$

arithmetic operations in $\mathbf{K}$.

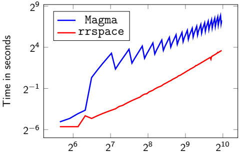
- Improves the complexity in $O(\deg(C)^6 \deg(D_+)^6)$ of the geometric algorithm of Huang and Ierardi.
- When $\deg(D_+) \leq \deg(C)^2$, complexity in $O(\deg(C)^{2\omega})$. Slightly improves Khuri-Makdisi in the special case of computing in Jacobians of smooth plane curves.
- Produces a Las Vegas algorithm at the cost of a small increase in complexity.

Experimental results

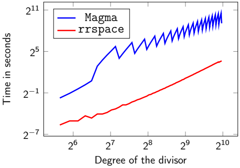
Comparison of the C++/NTL implementation rrspace and the Magma implementation RiemannRochSpace. Logarithmic scales.



Computation of a basis of $L(D)$ on a smooth curve of degree 10 on $GF(65521)$.



Computation of a basis of $L(D)$ on a nodal curve of degree 10 on $GF(65521)$.



Computation of a basis of $L(D)$ on a smooth curve of degree 10 on $GF(2^{32}-5)$.

Future works

- Structure of the linear systems ?
- What happens when the interpolating denominator encounters an unwanted singularity ?

Code available : https ://gitlab.inria.fr/pspaenle/rrspace
ArXiv link : https ://arxiv.org/abs/1811.08237

Future works

- Structure of the linear systems ?
- What happens when the interpolating denominator encounters an unwanted singularity ?

Code available : https ://gitlab.inria.fr/pspaenle/rrspace
ArXiv link : https ://arxiv.org/abs/1811.08237

Thank you !