Computing Coleman Integrals in SAGE

Robert Bradshaw and Kiran Kedlaya

SAGE Days 5: Oct 1, 2007

Properties of Coleman Integrals

Let $P, Q, R \in U = X - Z,\ \phi : X \to X'$, and $f \in A^{\dagger}$. Then the following properties hold:

Properties of Coleman Integrals

Let $P, Q, R \in U = X - Z$, $\phi : X \to X'$, and $f \in A^\dagger$. Then the following properties hold:

Properties

- Additivity $\int_P^Q \omega + \int_Q^R \omega = \int_P^R \omega$

Properties of Coleman Integrals

Let $P, Q, R \in U = X - Z,\ \phi : X \to X'$, and $f \in A^\dagger$. Then the following properties hold:

Properties

- Additivity $\int_P^Q \omega + \int_Q^R \omega = \int_P^R \omega$
- Linearity $\int_P^Q (\alpha\omega + \beta\eta) = \alpha \int_P^Q \omega + \beta \int_P^Q \eta$

Properties of Coleman Integrals

Let $P, Q, R \in U = X - Z,\ \phi : X \to X'$, and $f \in A^\dagger$. Then the following properties hold:

Properties

- Additivity $\int_P^Q \omega + \int_Q^R \omega = \int_P^R \omega$
- Linearity $\int_P^Q (\alpha\omega + \beta\eta) = \alpha \int_P^Q \omega + \beta \int_P^Q \eta$
- Change of variables $\int_{\phi(P)}^{\phi(Q)} \omega = \int_P^Q \phi^*(\omega)$

Properties of Coleman Integrals

Let $P, Q, R \in U = X - Z,\ \phi : X \to X'$, and $f \in A^\dagger$. Then the following properties hold:

Properties

- Additivity $\int_P^Q \omega + \int_Q^R \omega = \int_P^R \omega$
- Linearity $\int_P^Q (\alpha\omega + \beta\eta) = \alpha \int_P^Q \omega + \beta \int_P^Q \eta$
- Change of variables $\int_{\phi(P)}^{\phi(Q)} \omega = \int_P^Q \phi^*(\omega)$
- FTC $\int_P^Q df = f(Q) - f(P)$

Properties of Coleman Integrals

Let $P, Q, R \in U = X - Z$, $\phi : X \to X'$, and $f \in A^\dagger$. Then the following properties hold:

Properties

- Additivity $\quad \int_P^Q \omega + \int_Q^R \omega = \int_P^R \omega$
- Linearity $\quad \int_P^Q (\alpha\omega + \beta\eta) = \alpha \int_P^Q \omega + \beta \int_P^Q \eta$
- Change of variables $\quad \int_{\phi(P)}^{\phi(Q)} \omega = \int_P^Q \phi^*(\omega)$
- FTC $\quad \int_P^Q df = f(Q) - f(P)$
- Local analyticity (on each open disc of U^{an}).

Tiny Integrals

In each residue class of U the function is locally analytic.

Tiny Integrals

In each residue class of U the function is locally analytic.

- Let P, Q be in the same residue class.

Tiny Integrals

In each residue class of U the function is locally analytic.

- Let P, Q be in the same residue class.
- Choose a linear interpolation from P to Q.

Tiny Integrals

In each residue class of U the function is locally analytic.

- Let P, Q be in the same residue class.
- Choose a linear interpolation from P to Q.
  - E.g. if $y^2 = a(x)$ we can let

$$x(t) = tP_x + (1-t)Q_x \quad \text{and} \quad y(t) = \sqrt{a(x(t))}$$

Tiny Integrals

In each residue class of U the function is locally analytic.

- Let P, Q be in the same residue class.
- Choose a linear interpolation from P to Q.
  - E.g. if $y^2 = a(x)$ we can let

$$x(t) = tP_x + (1-t)Q_x \quad \text{and} \quad y(t) = \sqrt{a(x(t))}$$

- Formally integrate

$$\int_P^Q \omega = \int_P^Q f(x,y)\frac{dx}{y} = \int_0^1 \frac{f(x(t),y(t))}{y(t)}\frac{dx}{dt}dt$$

as a power series in t.

Tiny Integrals

In each residue class of U the function is locally analytic.

- Let P, Q be in the same residue class.
- Choose a linear interpolation from P to Q.
  - E.g. if $y^2 = a(x)$ we can let

$$x(t) = tP_x + (1-t)Q_x \quad \text{and} \quad y(t) = \sqrt{a(x(t))}$$

- Formally integrate

$$\int_P^Q \omega = \int_P^Q f(x,y)\frac{dx}{y} = \int_0^1 \frac{f(x(t),y(t))}{y(t)}\frac{dx}{dt}dt$$

  as a power series in t.
- Because P and Q are in the same residue class, all power series are actually power series in pt. This lets us calculate $\int_P^Q \omega$ to any desired precision *if* P and Q are in the same residue class.

Teichmüller Points

Let $\phi : A^\dagger \to A^\dagger$ be a q-power Frobenius lift viewed as acting on $A\dagger$.

Teichmüller Points

Let $\phi : A^\dagger \to A^\dagger$ be a q-power Frobenius lift viewed as acting on $A\dagger$.

Definition

A **Teichmüller point** is a point P such that $\phi(P) = P$.

Teichmüller Points

Let $\phi : A^\dagger \to A^\dagger$ be a q-power Frobenius lift viewed as acting on $A\dagger$.

Definition

A **Teichmüller point** is a point P such that $\phi(P) = P$.

Fact

There is a Teichmüller point in every residue class.

Kedlaya's algorithm for Computing the Action of Frobenius

Let $C : y^2 = f(x)$ where f is of degree $2g + 1$, and $\phi =$Frob$_p$. .

Kedlaya's algorithm for Computing the Action of Frobenius

Let $C : y^2 = f(x)$ where f is of degree $2g+1$, and $\phi =$Frob$_p$. .

- There is a (not necessarily canonical) lift of ϕ acting on $A^\dagger$, but we can extend it non-cannonically by letting $\phi(x) = x^p$ and

$$\phi(y) = \sqrt{\phi(f)(x^p)} = y^p \left(1 + \frac{\phi(f)(x^p) - f(x)^p}{y^{2p}}\right)^{1/2}.$$

Kedlaya's algorithm for Computing the Action of Frobenius

Let $C : y^2 = f(x)$ where f is of degree $2g+1$, and $\phi = \mathrm{Frob}_p$. .

- There is a (not necessarily canonical) lift of ϕ acting on $A^\dagger$, but we can extend it non-cannonically by letting $\phi(x) = x^p$ and

$$\phi(y) = \sqrt{\phi(f)(x^p)} = y^p \left(1 + \frac{\phi(f)(x^p) - f(x)^p}{y^{2p}}\right)^{1/2}.$$

- For each basis element $\frac{x^i dx}{y}$ of $H^1_{MW}(C)$ compute

$$\phi^*\left(\frac{x^i dx}{y}\right) = \frac{p x^{pi+p-1} dx}{\phi(y)}.$$

Kedlaya's algorithm for Computing the Action of Frobenius

Let $C : y^2 = f(x)$ where f is of degree $2g+1$, and $\phi =$Frob$_p$. .

- There is a (not necessarily canonical) lift of ϕ acting on $A^\dagger$, but we can extend it non-cannonically by letting $\phi(x) = x^p$ and

$$\phi(y) = \sqrt{\phi(f)(x^p)} = y^p \left(1 + \frac{\phi(f)(x^p) - f(x)^p}{y^{2p}}\right)^{1/2}.$$

- For each basis element $\frac{x^i dx}{y}$ of $H^1_{MW}(C)$ compute

$$\phi^*\left(\frac{x^i dx}{y}\right) = \frac{p x^{pi+p-1} dx}{\phi(y)}.$$

- Write each $\phi\left(x^i dx/y\right)$ in terms of $x^i dx/y$, $0 \le i < 2g$, using the relations $y^2 = f(x)$, $2ydy = f'(x)dx$ and $d(x^i y^j) = 0$.

Kedlaya's algorithm for Computing the Action of Frobenius

Let $C : y^2 = f(x)$ where f is of degree $2g+1$, and $\phi = \text{Frob}_p$. .

- There is a (not necessarily canonical) lift of ϕ acting on $A^\dagger$, but we can extend it non-cannonically by letting $\phi(x) = x^p$ and

$$\phi(y) = \sqrt{\phi(f)(x^p)} = y^p \left(1 + \frac{\phi(f)(x^p) - f(x)^p}{y^{2p}}\right)^{1/2}.$$

- For each basis element $\frac{x^i dx}{y}$ of $H^1_{MW}(C)$ compute

$$\phi^*\left(\frac{x^i dx}{y}\right) = \frac{p x^{pi+p-1} dx}{\phi(y)}.$$

- Write each $\phi\left(x^i dx/y\right)$ in terms of $x^i dx/y$, $0 \le i < 2g$, using the relations $y^2 = f(x)$, $2ydy = f'(x)dx$ and $d(x^i y^j) = 0$.
- As before, high powers of y are necessarily to be p-adicly small.

Connecting Integrals

Let P, Q be Teichmüller points.

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

$$\int_P^Q \omega_i = \int_P^Q \omega_i$$

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

$$\int_P^Q \omega_i = \int_{\phi(P)}^{\phi(Q)} \omega_i$$

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

$$\int_P^Q \omega_i = \int_P^Q \phi^* \omega_i$$

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

$$\int_P^Q \omega_i = \int_P^Q \left(df_i + \sum_{j=1}^{n} A_{ij} \omega_j \right)$$

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

$$\int_P^Q \omega_i = \int_P^Q df_i + \sum_{j=1}^{n} A_{ij} \int_P^Q \omega_j$$

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

$$\int_P^Q \omega_i = f_i(Q) - f_i(P) + \sum_{j=1}^{n} A_{ij} \int_P^Q \omega_j$$

Connecting Integrals

Let P, Q be Teichmüller points.

- Choose a basis $\omega_1, \ldots, \omega_n$ for $H^1_{MW}(\overline{U})$.
- Calculate the action of ϕ on each basis element, letting

$$\phi^* \omega_i = df_i + \sum_{j=1}^{n} A_{ij} \omega_j.$$

- Compute each $\int_P^Q \omega_j$ by solving a linear system

$$\int_P^Q \omega_i = f_i(Q) - f_i(P) + \sum_{j=1}^{n} A_{ij} \int_P^Q \omega_j$$

- Use linearity to integrate arbitrary ω.

Putting it all together

Let P, Q be arbitrary points in U.

Putting it all together

Let P, Q be arbitrary points in U.

- Compute Teichmüller points P', Q' in the same residue classes as P, Q respectively.

Putting it all together

Let P, Q be arbitrary points in U.

- Compute Teichmüller points P', Q' in the same residue classes as P, Q respectively.
- Now we can calculate

$$\int_P^Q \omega =$$

Putting it all together

Let P, Q be arbitrary points in U.

- Compute Teichmüller points P', Q' in the same residue classes as P, Q respectively.
- Now we can calculate

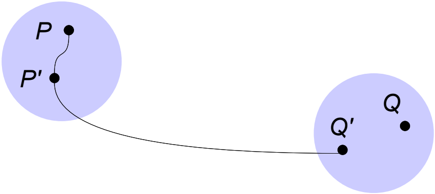
$$\int_P^Q \omega = \int_P^{P'} \omega$$

Putting it all together

Let P, Q be arbitrary points in U.

- Compute Teichmüller points P', Q' in the same residue classes as P, Q respectively.
- Now we can calculate

$$\int_P^Q \omega = \int_P^{P'} \omega + \int_{P'}^{Q'} \omega$$

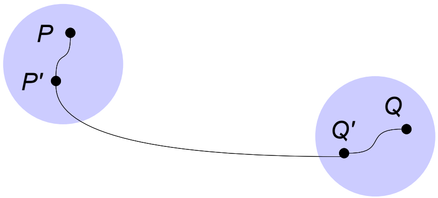


Putting it all together

Let P, Q be arbitrary points in U.

- Compute Teichmüller points P', Q' in the same residue classes as P, Q respectively.
- Now we can calculate

$$\int_P^Q \omega = \int_P^{P'} \omega + \int_{P'}^{Q'} \omega + \int_{Q'}^{Q} \omega$$



History in SAGE

- MSRI Graduate Student Workshop 2006
  - Kedlaya's algorithm for computing action of Frobenius implemented for Elliptic Curves in context of point counting and computing p-adic heights.
  - Robert Bradshaw, David Harvey, Jen Balakrishnan, Liang Xiao

History in SAGE

- MSRI Graduate Student Workshop 2006
  - Kedlaya's algorithm for computing action of Frobenius implemented for Elliptic Curves in context of point counting and computing p-adic heights.
  - Robert Bradshaw, David Harvey, Jen Balakrishnan, Liang Xiao
- Arizona Winter School 2007
  - Extend algorithm to keep track of exact forms, hyperelliptic curves
  - Teichmüller points, tiny integrals, etc.
  - Robert Bradshaw, Kiran Kedlaya, Ralf Gerkmann, Miljan Brakovevic
  - Much optimization since

History in SAGE

- MSRI Graduate Student Workshop 2006
  - Kedlaya's algorithm for computing action of Frobenius implemented for Elliptic Curves in context of point counting and computing p-adic heights.
  - Robert Bradshaw, David Harvey, Jen Balakrishnan, Liang Xiao
- Arizona Winter School 2007
  - Extend algorithm to keep track of exact forms, hyperelliptic curves
  - Teichmüller points, tiny integrals, etc.
  - Robert Bradshaw, Kiran Kedlaya, Ralf Gerkmann, Miljan Brakovevic
  - Much optimization since
- Spring 2007
  - David Harvey's asymptotic improvements
  - Hyperelliptic curve implementation in C++
    - Very fast, but unsuitable for Coleman Integrals. (Maybe not?)

Implementation Details

- Uses Newton iteration to calculate Teichmüller points, square roots, etc.

Implementation Details

- Uses Newton iteration to calculate Teichmüller points, square roots, etc.
- Specialized parent classes `SpecialCubicQuotientRing`, `SpecialHyperellipticQuotientRing`, and `MonskyWashnitzerDifferentialRing`.

Implementation Details

- Uses Newton iteration to calculate Teichmüller points, square roots, etc.
- Specialized parent classes `SpecialCubicQuotientRing`, `SpecialHyperellipticQuotientRing`, and `MonskyWashnitzerDifferentialRing`.
- Required and resulted in massive speedup of Laurent series and power series (among other things).

Implementation Details

Main files

```
$ wc -l ...

  2285   elliptic_curves/monsky_washnitzer.py
   182   elliptic_curves/ell_padic_field.py
   232   hyperelliptic_curves/hyperelliptic_padic_field.py
   131   hyperelliptic_curves/frobenius.pyx
  2015   hyperelliptic_curves/frobenius_cpp.cpp
```

Demo

Demo

```
sage: K = pAdicField(19, 15)
sage: E = EllipticCurve(K, '11a').weierstrass_model()
sage: P = E(K(14/3), K(11/2))
sage: 5*P
(0 : 1 + O(19^15) : 0)
sage: w = E.invariant_differential()
sage: w.coleman_integral(P, 2*P)
O(11^7)
```

Demo

```
sage: K = pAdicField(11, 7)
sage: x = polygen(K)
sage: C = HyperellipticCurve(x^5 + 33/16*x^4
          + 3/4*x^3 + 3/8*x^2 - 1/4*x + 1/16)
sage: P = C(-1, 1); P1 = C(-1, -1)
sage: Q = C(0, 1/4); Q1 = C(0, -1/4)
sage: x, y = C.monsky_washnitzer_gens()
sage: w = C.invariant_differential()
sage: w.coleman_integral(P, Q)
O(11^7)
```

What's taking it so long?

```
sage: w.coleman_integral(P, 2*P)

    0.000s -- setup
    0.210s -- tiny integrals
    1.307s -- mw calc
    0.000s -- eval f
    0.002s -- eval f Rational Field
    0.395s -- changing rings
    0.011s -- eval f 19-adic Field with capped relativ
    0.004s -- solve lin system
```

What's taking it so long?

```
sage: w.coleman_integral(P, 2*P)

    0.000s -- setup
    0.210s -- tiny integrals
    1.307s -- mw calc
    0.000s -- eval f
    0.002s -- eval f Rational Field
    0.395s -- changing rings
    0.011s -- eval f 19-adic Field with capped relativ
    0.004s -- solve lin system
```

What's taking it so long?

```
matrix_of_frobenius_hyperelliptic

    0.000s -- setup
    0.007s -- x_to_p
    0.005s -- frob_Q
    0.149s -- sqrt
    0.013s -- compose
    0.019s -- setup
    0.185s -- frob basis elements
    0.193s -- rationalize
    0.926s -- reduce
```

What's taking it so long?

- Currently reduce goes back and forth between $\mathbb{Q}$ and $\mathbb{Z}/p^n\mathbb{Z}$.

What's taking it so long?

- Currently reduce goes back and forth between $\mathbb{Q}$ and $\mathbb{Z}/p^n\mathbb{Z}$.
  - Linear algebra in $\mathbb{Q}$ is fast, but coefficients grow quickly.

What's taking it so long?

- Currently reduce goes back and forth between $\mathbb{Q}$ and $\mathbb{Z}/p^n\mathbb{Z}$.
  - Linear algebra in $\mathbb{Q}$ is fast, but coefficients grow quickly.
- This is the perfect application of fast p-adic linear algebra (and polynomials).

What's taking it so long?

- Currently reduce goes back and forth between $\mathbb{Q}$ and $\mathbb{Z}/p^n\mathbb{Z}$.
  - Linear algebra in $\mathbb{Q}$ is fast, but coefficients grow quickly.
- This is the perfect application of fast p-adic linear algebra (and polynomials).
  - We don't even need precision tacking

What's taking it so long?

- Currently reduce goes back and forth between $\mathbb{Q}$ and $\mathbb{Z}/p^n\mathbb{Z}$.
  - Linear algebra in $\mathbb{Q}$ is fast, but coefficients grow quickly.
- This is the perfect application of fast p-adic linear algebra (and polynomials).
  - We don't even need precision tacking
- There are other obvious optimizations elsewhere too

Future Work

- Iterated Coleman Integrals

Future Work

- Iterated Coleman Integrals
- Extend to non-prime $\mathbb{Q}_q$

Future Work

- Iterated Coleman Integrals
- Extend to non-prime $\mathbb{Q}_q$
- Extend to $p = 2, 3$

Future Work

- Iterated Coleman Integrals
- Extend to non-prime $\mathbb{Q}_q$
- Extend to $p = 2, 3$
- Optimize, convert to Cython or C/C++