Quantum complexity and L-functions

Kiran S. Kedlaya

Department of Mathematics, University of California San Diego
kedlaya@ucsd.edu
These slides can be downloaded from https://kskedlaya.org/slides/.

Quantum Algorithms in Number Theory
Fields Institute, Toronto, Canada (virtual)
April 21, 2022

Supported by NSF (grant DMS-2053473) and UC San Diego (Warschawski Professorship). Based in part on discussions with Diptajit Roy (IIT Kanpur).

The UC San Diego campus sits on unceded ancestral land of the Kumeyaay Nation. The Kumeyaay people continue to have an important and thriving presence in the region.

Contents

The Hasse-Weil zeta function of a variety

Let X be an algebraic variety of dimension d over a finite field $\mathbb{F}_q$. The **Hasse-Weil zeta function** is the power series*

$$Z(X, T) = \exp\left(\sum_{n=1}^{\infty} \frac{T^n}{n} \#X(\mathbb{F}_{q^n})\right).$$

It is known that:

- $Z(X, T) \in 1 + T\mathbb{Z}[[T]]$;
- $Z(X, T)$ represents a rational function in T;
- every zero or pole z of $Z(X, T)$ in $\mathbb{C}$ satisfies $|z| = q^{-i/2}$ for some $i \in \{0, \ldots, 2d\}$;
- if X is smooth proper, $Z(X, q^{-d}T^{-1}) = T^* q^* Z(X, T)$.

*To make it look like more like the Riemann zeta function, take $T = q^{-s}$.

The Hasse-Weil zeta function of a variety

Let X be an algebraic variety of dimension d over a finite field $\mathbb{F}_q$. The **Hasse-Weil zeta function** is the power series*

$$Z(X,T) = \exp\left(\sum_{n=1}^{\infty} \frac{T^n}{n} \#X(\mathbb{F}_{q^n})\right).$$

It is known that:

- $Z(X,T) \in 1 + T\mathbb{Z}[[T]]$;
- $Z(X,T)$ represents a rational function in T;
- every zero or pole z of $Z(X,T)$ in $\mathbb{C}$ satisfies $|z| = q^{-i/2}$ for some $i \in \{0,\dots,2d\}$;
- if X is smooth proper, $Z(X, q^{-d}T^{-1}) = T^* q^* Z(X,T)$.

*To make it look like more like the Riemann zeta function, take $T = q^{-s}$.

The Hasse-Weil zeta function of a variety

Let X be an algebraic variety of dimension d over a finite field $\mathbb{F}_q$. The **Hasse-Weil zeta function** is the power series*

$$Z(X,T) = \exp\left(\sum_{n=1}^{\infty} \frac{T^n}{n} \#X(\mathbb{F}_{q^n})\right).$$

It is known that:

- $Z(X,T) \in 1 + T\mathbb{Z}[[T]]$;
- $Z(X,T)$ represents a rational function in T;
- every zero or pole z of $Z(X,T)$ in $\mathbb{C}$ satisfies $|z| = q^{-i/2}$ for some $i \in \{0,\dots,2d\}$;
- if X is smooth proper, $Z(X, q^{-d}T^{-1}) = T^* q^* Z(X,T)$.

*To make it look like more like the Riemann zeta function, take $T = q^{-s}$.

The Hasse-Weil zeta function of a variety

Let X be an algebraic variety of dimension d over a finite field $\mathbb{F}_q$. The **Hasse-Weil zeta function** is the power series*

$$Z(X, T) = \exp\left(\sum_{n=1}^{\infty} \frac{T^n}{n} \#X(\mathbb{F}_{q^n})\right).$$

It is known that:

- $Z(X, T) \in 1 + T\mathbb{Z}[[T]]$;
- $Z(X, T)$ represents a rational function in T;
- every zero or pole z of $Z(X, T)$ in $\mathbb{C}$ satisfies $|z| = q^{-i/2}$ for some $i \in \{0, \dots, 2d\}$;
- if X is smooth proper, $Z(X, q^{-d}T^{-1}) = T^* q^* Z(X, T)$.

*To make it look like more like the Riemann zeta function, take $T = q^{-s}$.

The Hasse-Weil zeta function of a variety

Let X be an algebraic variety of dimension d over a finite field $\mathbb{F}_q$. The **Hasse-Weil zeta function** is the power series*

$$Z(X,T) = \exp\left(\sum_{n=1}^{\infty} \frac{T^n}{n} \#X(\mathbb{F}_{q^n})\right).$$

It is known that:

- $Z(X,T) \in 1 + T\mathbb{Z}[[T]]$;
- $Z(X,T)$ represents a rational function in T;
- every zero or pole z of $Z(X,T)$ in $\mathbb{C}$ satisfies $|z| = q^{-i/2}$ for some $i \in \{0,\dots,2d\}$;
- if X is smooth proper, $Z(X, q^{-d}T^{-1}) = T^* q^* Z(X,T)$.

*To make it look like more like the Riemann zeta function, take $T = q^{-s}$.

Examples

For $X = \mathbb{P}^n$,

$$Z(X,T) = \frac{1}{(1-T)(1-qT)\cdots(1-q^nT)}.$$

For X an elliptic curve,

$$Z(X,T) = \frac{1+aT+qT^2}{(1-T)(1-qT)}$$

where $|a| \leq 2q^{1/2}$. Note that $\#X(\mathbb{F}_q) = q+1+a$.

For X a curve of genus g,

$$Z(X,T) = \frac{1+a_1T+\cdots+a_gT^g+qa_{g-1}T^{g+1}+\cdots+q^gT^{2g}}{(1-T)(1-qT)}.$$

Examples

For $X = \mathbb{P}^n$,

$$Z(X, T) = \frac{1}{(1-T)(1-qT)\cdots(1-q^nT)}.$$

For X an elliptic curve,

$$Z(X, T) = \frac{1 + aT + qT^2}{(1-T)(1-qT)}$$

where $|a| \leq 2q^{1/2}$. Note that $\#X(\mathbb{F}_q) = q + 1 + a$.

For X a curve of genus g,

$$Z(X, T) = \frac{1 + a_1T + \cdots + a_gT^g + qa_{g-1}T^{g+1} + \cdots + q^gT^{2g}}{(1-T)(1-qT)}.$$

Examples

For $X = \mathbb{P}^n$,

$$Z(X,T) = \frac{1}{(1-T)(1-qT)\cdots(1-q^nT)}.$$

For X an elliptic curve,

$$Z(X,T) = \frac{1+aT+qT^2}{(1-T)(1-qT)}$$

where $|a| \leq 2q^{1/2}$. Note that $\#X(\mathbb{F}_q) = q+1+a$.

For X a curve of genus g,

$$Z(X,T) = \frac{1+a_1T+\cdots+a_gT^g+qa_{g-1}T^{g+1}+\cdots+q^gT^{2g}}{(1-T)(1-qT)}.$$

A word of warning

It is an important algorithmic problem to compute $Z(X, T)$ from X. However, one has to be careful in the formulation: in terms of the length of a **sparse** representation of X, the problem is NP-complete because it includes 3-SAT (taking $q = 2$).

For this reason, I usually take d to be **fixed**. This leaves me free to vary:

- the cardinality q of the base field, and
- the "arithmetic complexity" of X. For example, if we require X to be an affine hypersurface, we can use the **degree** of the defining polynomial. If X is a smooth projective curve, it is natural to use its **genus**.

A word of warning

It is an important algorithmic problem to compute $Z(X, T)$ from X. However, one has to be careful in the formulation: in terms of the length of a **sparse** representation of X, the problem is NP-complete because it includes 3-SAT (taking $q = 2$).

For this reason, I usually take d to be **fixed**. This leaves me free to vary:

- the cardinality q of the base field, and
- the "arithmetic complexity" of X. For example, if we require X to be an affine hypersurface, we can use the **degree** of the defining polynomial. If X is a smooth projective curve, it is natural to use its **genus**.

A word of warning

It is an important algorithmic problem to compute $Z(X, T)$ from X. However, one has to be careful in the formulation: in terms of the length of a **sparse** representation of X, the problem is NP-complete because it includes 3-SAT (taking $q = 2$).

For this reason, I usually take d to be **fixed**. This leaves me free to vary:

- the cardinality q of the base field, and
- the "arithmetic complexity" of X. For example, if we require X to be an affine hypersurface, we can use the **degree** of the defining polynomial. If X is a smooth projective curve, it is natural to use its **genus**.

A word of warning

It is an important algorithmic problem to compute $Z(X, T)$ from X. However, one has to be careful in the formulation: in terms of the length of a **sparse** representation of X, the problem is NP-complete because it includes 3-SAT (taking $q = 2$).

For this reason, I usually take d to be **fixed**. This leaves me free to vary:

- the cardinality q of the base field, and
- the "arithmetic complexity" of X. For example, if we require X to be an affine hypersurface, we can use the **degree** of the defining polynomial. If X is a smooth projective curve, it is natural to use its **genus**.

Some classical complexity results

For simplicity, assume that X is an affine hypersurface, and take all classical algorithms to be randomized.

Theorem (Lauder–Wan)

There is a classical algorithm to compute $Z(X, T)$ from X in time polynomial in p, $\deg X$, and $\log_p q$.

Theorem (Schoof–Pila)

For $d = 1$ and $\deg X$ fixed, there is a classical algorithm to compute $Z(X, T)$ from X in time polynomial in $\log q$.

Theorem (Harvey)

*For **fixed** X of any dimension, there is a classical algorithm to compute $Z(X, T)$ from X in time polynomial in $\log q$.*

A quantum complexity result

Theorem (K)

For $d = 1$, there is a quantum algorithm to compute $Z(X, T)$ in time polynomial in $\deg X$ and $\log q$.

The point is that there are some abelian groups related to $Z(X, T)$: the groups of $\mathbb{F}_{q^n}$-rational points of the **Jacobian variety** $J(X)$. If we write

$$Z(X, T) = \frac{P(T)}{(1 - T)(1 - qT)}, \text{ then } \#J(X)(\mathbb{F}_{q^n}) = P(1)P(\zeta_n) \cdots P(\zeta_n^{n-1}).$$

We exhibit $J(X)$ as a black box group using divisor classes, use Shor to compute $\#J(X)(\mathbb{F}_{q^n})$ for $n = 1, \dots, O(\deg X)$, and recover P from these.

A quantum complexity result

Theorem (K)

For $d = 1$, there is a quantum algorithm to compute $Z(X, T)$ in time polynomial in $\deg X$ and $\log q$.

The point is that there are some abelian groups related to $Z(X, T)$: the groups of $\mathbb{F}_{q^n}$-rational points of the **Jacobian variety** $J(X)$. If we write

$$Z(X, T) = \frac{P(T)}{(1-T)(1-qT)}, \text{ then } \#J(X)(\mathbb{F}_{q^n}) = P(1)P(\zeta_n)\cdots P(\zeta_n^{n-1}).$$

We exhibit $J(X)$ as a black box group using divisor classes, use Shor to compute $\#J(X)(\mathbb{F}_{q^n})$ for $n = 1, \ldots, O(\deg X)$, and recover P from these.

A quantum complexity result

Theorem (K)

For $d = 1$, there is a quantum algorithm to compute $Z(X, T)$ in time polynomial in $\deg X$ and $\log q$.

The point is that there are some abelian groups related to $Z(X, T)$: the groups of $\mathbb{F}_{q^n}$-rational points of the **Jacobian variety** $J(X)$. If we write

$$Z(X, T) = \frac{P(T)}{(1 - T)(1 - qT)}, \text{ then } \#J(X)(\mathbb{F}_{q^n}) = P(1)P(\zeta_n) \cdots P(\zeta_n^{n-1}).$$

We exhibit $J(X)$ as a black box group using divisor classes, use Shor to compute $\#J(X)(\mathbb{F}_{q^n})$ for $n = 1, \ldots, O(\deg X)$, and recover P from these.

The role of cohomology

Most knowledge about $Z(X, T)$ comes from a **cohomological interpretation** arising from the **Lefschetz trace formula**:

$$Z(X, T) = \prod_{i=0}^{2d} \det(1 - TF, H^i(X))^{(-1)^{i+1}}$$

where $H^i(X)$ is a certain finite-dimensional vector space over some field K of characteristic 0 and F is some linear operator ("Frobenius") acting on each $H^i(X)$. The two known approaches:

- **étale cohomology**: $K = \mathbb{Q}_\ell$ for some prime $\ell \neq p$. In practice, we instead use $K = \mathbb{F}_\ell$ for "enough" small ℓ to pin down $Z(X, T)$.
- **crystalline cohomology** and related constructions: K is a finite extension of $\mathbb{Q}_p$.

The role of cohomology

Most knowledge about $Z(X, T)$ comes from a **cohomological interpretation** arising from the **Lefschetz trace formula**:

$$Z(X, T) = \prod_{i=0}^{2d} \det(1 - TF, H^i(X))^{(-1)^{i+1}}$$

where $H^i(X)$ is a certain finite-dimensional vector space over some field K of characteristic 0 and F is some linear operator ("Frobenius") acting on each $H^i(X)$. The two known approaches:

- **étale cohomology**: $K = \mathbb{Q}_\ell$ for some prime $\ell \neq p$. In practice, we instead use $K = \mathbb{F}_\ell$ for "enough" small ℓ to pin down $Z(X, T)$.
- **crystalline cohomology** and related constructions: K is a finite extension of $\mathbb{Q}_p$.

The role of cohomology

Most knowledge about $Z(X, T)$ comes from a **cohomological interpretation** arising from the **Lefschetz trace formula**:

$$Z(X, T) = \prod_{i=0}^{2d} \det(1 - TF, H^i(X))^{(-1)^{i+1}}$$

where $H^i(X)$ is a certain finite-dimensional vector space over some field K of characteristic 0 and F is some linear operator ("Frobenius") acting on each $H^i(X)$. The two known approaches:

- **étale cohomology**: $K = \mathbb{Q}_\ell$ for some prime $\ell \neq p$. In practice, we instead use $K = \mathbb{F}_\ell$ for "enough" small ℓ to pin down $Z(X, T)$.
- **crystalline cohomology** and related constructions: K is a finite extension of $\mathbb{Q}_p$.

Crystalline cohomology

Crystalline cohomology groups can be defined in various ways, some of which **are** intrinsically computable. For example, the interpretation in terms of **Monsky–Washnitzer cohomology** is used in many practical algorithms in PARI, Sage, Magma, etc.

However, it seems unavoidable for these constructions to incur polynomial (at least square-root) dependence on p, unless you amortize over many primes (Harvey, Harvey–Sutherland).

A closely related problem: given a power series over $\mathbb{Q}$ satisfying a fixed ODE, compute the p-th coefficient modulo p in time polynomial in $\log p$.

Crystalline cohomology

Crystalline cohomology groups can be defined in various ways, some of which **are** intrinsically computable. For example, the interpretation in terms of **Monsky–Washnitzer cohomology** is used in many practical algorithms in PARI, Sage, Magma, etc.

However, it seems unavoidable for these constructions to incur polynomial (at least square-root) dependence on p, unless you amortize over many primes (Harvey, Harvey–Sutherland).

A closely related problem: given a power series over $\mathbb{Q}$ satisfying a fixed ODE, compute the p-th coefficient modulo p in time polynomial in $\log p$.

Crystalline cohomology

Crystalline cohomology groups can be defined in various ways, some of which **are** intrinsically computable. For example, the interpretation in terms of **Monsky–Washnitzer cohomology** is used in many practical algorithms in PARI, Sage, Magma, etc.

However, it seems unavoidable for these constructions to incur polynomial (at least square-root) dependence on p, unless you amortize over many primes (Harvey, Harvey–Sutherland).

A closely related problem: given a power series over $\mathbb{Q}$ satisfying a fixed ODE, compute the p-th coefficient modulo p in time polynomial in $\log p$.

Étale cohomology and quantum Schoof–Pila?

The étale cohomology groups $H^i(X)$ are **not** defined in an intrinsically computable way. For $i = 1$, one can obtain a computable model using Jacobians; this is the basis of the quantum algorithm from earlier. It is also the basis of Schoof–Pila, but using $\mathbb{F}_\ell$-coefficients for a few small primes ℓ.

There is no analogue for $i > 1$, but when using $\mathbb{F}_\ell$ coefficients one can relate $H^i(X)$ to $H^1(Y)$ for a suitable curve Y. The catch is that the complexity of Y depends polynomially on ℓ, which breaks Schoof–Pila...

...but represents Frobenius as a “black box linear transformation” over $\mathbb{F}_\ell$. The charpoly can be found via a hidden subgroup problem (Shor–Kitaev)!

Challenge: turn this sketch into an explicit quantum algorithm to compute $Z(X, T)$ from X in time polynomial in $\log q$ and $\deg X$. The computability of étale cohomology is a theorem of Madore–Orgogozo, but with no complexity analysis.

Étale cohomology and quantum Schoof–Pila?

The étale cohomology groups $H^i(X)$ are **not** defined in an intrinsically computable way. For $i = 1$, one can obtain a computable model using Jacobians; this is the basis of the quantum algorithm from earlier. It is also the basis of Schoof–Pila, but using $\mathbb{F}_\ell$-coefficients for a few small primes ℓ.

There is no analogue for $i > 1$, but when using $\mathbb{F}_\ell$ coefficients one can relate $H^i(X)$ to $H^1(Y)$ for a suitable curve Y. The catch is that the complexity of Y depends polynomially on ℓ, which breaks Schoof–Pila...

...but represents Frobenius as a "black box linear transformation" over $\mathbb{F}_\ell$. The charpoly can be found via a hidden subgroup problem (Shor–Kitaev)!

Challenge: turn this sketch into an explicit quantum algorithm to compute $Z(X, T)$ from X in time polynomial in $\log q$ and $\deg X$. The computability of étale cohomology is a theorem of Madore–Orgogozo, but with no complexity analysis.

Étale cohomology and quantum Schoof–Pila?

The étale cohomology groups $H^i(X)$ are **not** defined in an intrinsically computable way. For $i = 1$, one can obtain a computable model using Jacobians; this is the basis of the quantum algorithm from earlier. It is also the basis of Schoof–Pila, but using $\mathbb{F}_\ell$-coefficients for a few small primes ℓ.

There is no analogue for $i > 1$, but when using $\mathbb{F}_\ell$ coefficients one can relate $H^i(X)$ to $H^1(Y)$ for a suitable curve Y. The catch is that the complexity of Y depends polynomially on ℓ, which breaks Schoof–Pila...

...but represents Frobenius as a "black box linear transformation" over $\mathbb{F}_\ell$. The charpoly can be found via a hidden subgroup problem (Shor–Kitaev)!

Challenge: turn this sketch into an explicit quantum algorithm to compute $Z(X, T)$ from X in time polynomial in $\log q$ and $\deg X$. The computability of étale cohomology is a theorem of Madore–Orgogozo, but with no complexity analysis.

Étale cohomology and quantum Schoof–Pila?

The étale cohomology groups $H^i(X)$ are **not** defined in an intrinsically computable way. For $i = 1$, one can obtain a computable model using Jacobians; this is the basis of the quantum algorithm from earlier. It is also the basis of Schoof–Pila, but using $\mathbb{F}_\ell$-coefficients for a few small primes ℓ.

There is no analogue for $i > 1$, but when using $\mathbb{F}_\ell$ coefficients one can relate $H^i(X)$ to $H^1(Y)$ for a suitable curve Y. The catch is that the complexity of Y depends polynomially on ℓ, which breaks Schoof–Pila...

...but represents Frobenius as a "black box linear transformation" over $\mathbb{F}_\ell$. The charpoly can be found via a hidden subgroup problem (Shor–Kitaev)!

Challenge: turn this sketch into an explicit quantum algorithm to compute $Z(X, T)$ from X in time polynomial in $\log q$ and $\deg X$. The computability of étale cohomology is a theorem of Madore–Orgogozo, but with no complexity analysis.

Contents

Modular forms

A **modular form of weight** k is a holomorphic function $f(z)$ for $\mathrm{Real}(z) > 0$ satisfying (a growth condition plus) the functional equation

$$f\left(\frac{az+b}{cz+d}\right) = (cz+d)^k f(z)$$

for matrices $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ over $\mathbb{Z}$ congruent to 1 modulo N (the **level**).

For given N and k, the modular forms of weight k form a finite-dimensional vector space. Since a modular form is invariant under $z \mapsto z + N$, it admits a Fourier expansion

$$f(z) = \sum_{n=0}^{\infty} a_{n/N}(f) q^{n/N}, \qquad q = e^{2\pi i z}.$$

Modular forms

A **modular form of weight** k is a holomorphic function $f(z)$ for $\mathrm{Real}(z) > 0$ satisfying (a growth condition plus) the functional equation

$$f\left(\frac{az+b}{cz+d}\right) = (cz+d)^k f(z)$$

for matrices $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ over $\mathbb{Z}$ congruent to 1 modulo N (the **level**).

For given N and k, the modular forms of weight k form a finite-dimensional vector space. Since a modular form is invariant under $z \mapsto z + N$, it admits a Fourier expansion

$$f(z) = \sum_{n=0}^{\infty} a_{n/N}(f) q^{n/N}, \qquad q = e^{2\pi i z}.$$

Fourier coefficients of modular forms

Theorem (Couveignes–Edixhoven)

Let f be a fixed modular form of weight $k \geq 2$ with coefficients in some number field. Under GRH (for Dedekind L-functions), there exists a classical algorithm which, given a **factored** *positive integer N, computes the N-th Fourier coefficient of f in time polynomial in $\log N$. (This immediately yields a quantum algorithm for N unfactored.)*

The proof is **very** similar to Schoof–Pila: one computes the mod-ℓ Galois representation associated to f for $O(\log N)$ different primes ℓ.

I do not know how the complexity behaves as you vary f.

Fourier coefficients of modular forms

Theorem (Couveignes–Edixhoven)

Let f be a fixed modular form of weight $k \geq 2$ with coefficients in some number field. Under GRH (for Dedekind L-functions), there exists a classical algorithm which, given a **factored** *positive integer N, computes the N-th Fourier coefficient of f in time polynomial in $\log N$. (This immediately yields a quantum algorithm for N unfactored.)*

The proof is **very** similar to Schoof–Pila: one computes the mod-ℓ Galois representation associated to f for $O(\log N)$ different primes ℓ.

I do not know how the complexity behaves as you vary f.

Fourier coefficients of modular forms

Theorem (Couveignes–Edixhoven)

Let f be a fixed modular form of weight $k \geq 2$ with coefficients in some number field. Under GRH (for Dedekind L-functions), there exists a classical algorithm which, given a **factored** *positive integer N, computes the N-th Fourier coefficient of f in time polynomial in $\log N$. (This immediately yields a quantum algorithm for N unfactored.)*

The proof is **very** similar to Schoof–Pila: one computes the mod-ℓ Galois representation associated to f for $O(\log N)$ different primes ℓ.

I do not know how the complexity behaves as you vary f.

Modular forms of weight 1

One can also consider modular forms of weight 1. For these one cannot interpret the Fourier coefficients **directly** in terms of Galois representations...

... but we can give such an interpretation using **congruences** between modular forms, to get an efficient classical algorithm...

... or interpret the coefficients in terms of orders of class groups of imaginary quadratic fields, to get an efficient quantum algorithm.

Modular forms of weight 1

One can also consider modular forms of weight 1. For these one cannot interpret the Fourier coefficients **directly** in terms of Galois representations...

... but we can give such an interpretation using **congruences** between modular forms, to get an efficient classical algorithm...

... or interpret the coefficients in terms of orders of class groups of imaginary quadratic fields, to get an efficient quantum algorithm.

Modular forms of weight 1

One can also consider modular forms of weight 1. For these one cannot interpret the Fourier coefficients **directly** in terms of Galois representations...

... but we can give such an interpretation using **congruences** between modular forms, to get an efficient classical algorithm...

... or interpret the coefficients in terms of orders of class groups of imaginary quadratic fields, to get an efficient quantum algorithm.

Half-integral weight

One can also consider modular forms of **half-integral weight**. For example, modular forms of weight $3/2$ arise as theta series of ternary quadratic forms.

There does **not** exist an interpretation of these Fourier coefficients in terms of Galois representations. Can these nonetheless be computed efficiently with a quantum algorithm, e.g., by expressing them as short sums of class numbers?

As a corollary, this would give a conditional (assuming **BSD**, the conjecture of Birch and Swinnerton–Dyer) quantum polynomial time algorithm to determine whether a given positive integer N is a **congruent number**, i.e., the area of a right triangle with rational side lengths.

Half-integral weight

One can also consider modular forms of **half-integral weight**. For example, modular forms of weight $3/2$ arise as theta series of ternary quadratic forms.

There does **not** exist an interpretation of these Fourier coefficients in terms of Galois representations. Can these nonetheless be computed efficiently with a quantum algorithm, e.g., by expressing them as short sums of class numbers?

As a corollary, this would give a conditional (assuming **BSD**, the conjecture of Birch and Swinnerton–Dyer) quantum polynomial time algorithm to determine whether a given positive integer N is a **congruent number**, i.e., the area of a right triangle with rational side lengths.

Half-integral weight

One can also consider modular forms of **half-integral weight**. For example, modular forms of weight $3/2$ arise as theta series of ternary quadratic forms.

There does **not** exist an interpretation of these Fourier coefficients in terms of Galois representations. Can these nonetheless be computed efficiently with a quantum algorithm, e.g., by expressing them as short sums of class numbers?

As a corollary, this would give a conditional (assuming **BSD**, the conjecture of Birch and Swinnerton–Dyer) quantum polynomial time algorithm to determine whether a given positive integer N is a **congruent number**, i.e., the area of a right triangle with rational side lengths.

The L-function of a modular form

Given a modular form $f = \sum_{n=1}^{\infty} a_n q^n$ of weight k, the associated L-**function** is the Dirichlet series

$$L(s, f) = \sum_{n=1}^{\infty} a_n n^{-s};$$

this converges absolutely for $\mathrm{Real}\, s > (k+1)/2$ and admits an analytic continuation to all of $\mathbb{C}$.

Much effort has been put into numerical computation of values of $L(s, f)$, particularly on the axis of symmetry $\mathrm{Real}\, s = k/2$ (Turing, ..., Booker, Platt). This computation includes a discrete Fourier transform†; does the use of quantum algorithms lead to any improvement?

†This is ultimately because the L-function is obtained from f by an integral transform (the **Mellin transform**).

The L-function of a modular form

Given a modular form $f = \sum_{n=1}^{\infty} a_n q^n$ of weight k, the associated L-**function** is the Dirichlet series

$$L(s, f) = \sum_{n=1}^{\infty} a_n n^{-s};$$

this converges absolutely for $\mathrm{Real}\, s > (k+1)/2$ and admits an analytic continuation to all of $\mathbb{C}$.

Much effort has been put into numerical computation of values of $L(s, f)$, particularly on the axis of symmetry $\mathrm{Real}\, s = k/2$ (Turing, ..., Booker, Platt). This computation includes a discrete Fourier transform[†]; does the use of quantum algorithms lead to any improvement?

[†] This is ultimately because the L-function is obtained from f by an integral transform (the **Mellin transform**).

Contents

The Mordell–Weil theorem

Theorem (Mordell for $K = \mathbb{Q}$, Weil in general)

Let E be an elliptic curve over a number field K. Then $E(K)$ is a finitely generated abelian group.

Warning: At present there is **no known unconditional algorithm** to compute $E(K)$ or even its rank. Under BSD‡, one can proceed by alternating a search for rational points (to get a lower bound on the rank) with a computation of derivatives of the L-function (to get an upper bound).

‡One must also assume the modularity of E, but this is known when K is a totally real field or a CM field.

The Mordell–Weil theorem

Theorem (Mordell for $K = \mathbb{Q}$, Weil in general)

Let E be an elliptic curve over a number field K. Then $E(K)$ is a finitely generated abelian group.

Warning: At present there is **no known unconditional algorithm** to compute $E(K)$ or even its rank. Under BSD‡, one can proceed by alternating a search for rational points (to get a lower bound on the rank) with a computation of derivatives of the L-function (to get an upper bound).

‡One must also assume the modularity of E, but this is known when K is a totally real field or a CM field.

Generators of Mordell–Weil

Let E be an elliptic curve over a number field K for which it is known that $\operatorname{rank} E(K) = 1$ (e.g., via a known case of BSD). Can one efficiently compute a generator of $E(K)$ in time polynomial in E? (This means polynomial in $\log \Delta_K$ and the logarithmic heights of the coefficients of E.)

In some sense this question has an information-theoretic negative answer; it can happen that the generators of $E(K)$ have too many digits in their projective coordinates. However, it may be possible to describe a **compact representation** of such a generator, e.g., by using large multiples of small points defined over a small extension field.[§]

[§] In some cases, finding this compact representation may be reducible to a hidden subgroup problem for $\mathbb{Z}^n$ (Kuperberg).

Generators of Mordell–Weil

Let E be an elliptic curve over a number field K for which it is known that $\operatorname{rank} E(K) = 1$ (e.g., via a known case of BSD). Can one efficiently compute a generator of $E(K)$ in time polynomial in E? (This means polynomial in $\log \Delta_K$ and the logarithmic heights of the coefficients of E.)

In some sense this question has an information-theoretic negative answer; it can happen that the generators of $E(K)$ have too many digits in their projective coordinates. However, it may be possible to describe a **compact representation** of such a generator, e.g., by using large multiples of small points defined over a small extension field.[§]

[§] In some cases, finding this compact representation may be reducible to a hidden subgroup problem for $\mathbb{Z}^n$ (Kuperberg).

An analogous problem

Let $K = \mathbb{Q}(\sqrt{D})$ be a real quadratic number field. The group $\mathfrak{o}_K^\times$ has rank 1, generated modulo torsion by a **fundamental unit**; this is a solution of the Brahmagupta–Bhāskara–Brouncker(–Pell) equation

$$x^2 - Dy^2 = \pm 1.$$

In general, $\log x$ and $\log y$ can be as large as polynomial in D, so one cannot even write them in time polynomial in $\log D$.

However, one can express the fundamental unit as a product

$$\alpha_1^{e_1} \cdots \alpha_n^{e_n}$$

where $\alpha_1, \dots, \alpha_n \in \mathfrak{o}_K$ are **not necessarily units**. In this sense, there is a quantum algorithm to compute the fundamental unit (Hallgren), or more generally $\mathfrak{o}_K^\times$ for any number field K (Schmidt–Völlmer).

An analogous problem

Let $K = \mathbb{Q}(\sqrt{D})$ be a real quadratic number field. The group $\mathfrak{o}_K^\times$ has rank 1, generated modulo torsion by a **fundamental unit**; this is a solution of the Brahmagupta–Bhāskara–Brouncker(–Pell) equation

$$x^2 - Dy^2 = \pm 1.$$

In general, $\log x$ and $\log y$ can be as large as polynomial in D, so one cannot even write them in time polynomial in $\log D$.

However, one can express the fundamental unit as a product

$$\alpha_1^{e_1} \cdots \alpha_n^{e_n}$$

where $\alpha_1, \ldots, \alpha_n \in \mathfrak{o}_K$ are **not necessarily units**. In this sense, there is a quantum algorithm to compute the fundamental unit (Hallgren), or more generally $\mathfrak{o}_K^\times$ for any number field K (Schmidt–Völlmer).

Heegner points

When E is defined over $\mathbb{Q}$ and has analytic rank 1 (i.e., the L-function vanishes to order 1 at $s = 1$), one can generate points of $E(\mathbb{Q})$ by taking traces of CM points on modular Jacobians. This is in the spirit of a compact representation, except that:

- the dimension of the Jacobian depends on the conductor of E, which is exponentially large;
- the CM points are defined over fields whose degree is also exponentially large.

Heegner points

When E is defined over $\mathbb{Q}$ and has analytic rank 1 (i.e., the L-function vanishes to order 1 at $s = 1$), one can generate points of $E(\mathbb{Q})$ by taking traces of CM points on modular Jacobians. This is in the spirit of a compact representation, except that:

- the dimension of the Jacobian depends on the conductor of E, which is exponentially large;
- the CM points are defined over fields whose degree is also exponentially large.

Heegner points

When E is defined over $\mathbb{Q}$ and has analytic rank 1 (i.e., the L-function vanishes to order 1 at $s = 1$), one can generate points of $E(\mathbb{Q})$ by taking traces of CM points on modular Jacobians. This is in the spirit of a compact representation, except that:

- the dimension of the Jacobian depends on the conductor of E, which is exponentially large;
- the CM points are defined over fields whose degree is also exponentially large.

Alternate sources of compact representations?

Is there another way to represent the group $E(K)$ that might yield a compact representation?

Motivation: Beilinson has constructed certain elements in algebraic K-theory (namely $K_2(E)$) which were used by Kato to prove BSD for E of analytic rank 0. Can one do something similar when the analytic rank is 1?

Alternate sources of compact representations?

Is there another way to represent the group $E(K)$ that might yield a compact representation?

Motivation: Beilinson has constructed certain elements in algebraic K-theory (namely $K_2(E)$) which were used by Kato to prove BSD for E of analytic rank 0. Can one do something similar when the analytic rank is 1?