Computing Theta functions in quasi-optimal time

Hugo Labrande (joint with E. Thomé)

Séminaire LFant, Bordeaux, Jun 7th 2016

```
/* CARAMEL */                                          C,A,
/*         */                                        R,a,
/*         */                                      M,E,
                                                  L,i=
                                                 5,e,
   d[5],Q[999                  ]={0};main(N      ){for
  (;i--;e=scanf("%"           "d",d+i));for(A    =*d;
 ++i<A          ;++Q[        i*i%       A],R=  i[Q]?
R:i);             for(;i    --;)         for(M =A;M
--;N              +=!M*Q   [E%A           ],e+= Q[(A
+E*E-             R*L*      L%A)          %A])  for(
  E=i,L=M,a=4;a;C=           i*E+R*M*L,L=(M*E    +i*L)
    %A,E=C%A+a                --[d]);printf      ("%d"
                                                  "\n",
                                                   (e+N*
                                                     N)/2
  /* cc caramel.c; echo f3 f2 f1 f0 p | ./a.out */   -A);}
```

SECURITY | PRIVACY | ASSURANCE
at the University of Calgary

TL;DR

$\boldsymbol{\theta} : z \in \mathbb{C}^g,\ \tau \in M_g(\mathbb{C})$ sym. with $\mathrm{Im}(\tau) > 0 \quad \mapsto \theta(z, \tau) \in \mathbb{C}$

Important function (number theory, differential equations, complex Riemann surfaces...).

Naive algorithm: $O(\mathcal{M}(P)P^{g/2})$ bit operations to get a result accurate to P bits.

Contributions

New algorithm in $O(\mathcal{M}(P)\log P)$ (*quasi-optimal complexity*) for $g = 1, 2$ (most common cases in crypto).

Hints for genus g generalization in $O(2^g\mathcal{M}(P)\log P)$.

Beats naive algo after 300000 ($g = 1$) and 3000 digits ($g = 2$).

Gory details (precision loss, etc): ia.cr/2015/1104 & ia.cr/2016/179.

Outline

1. Genus 1 theta function
   - Applications
   - Naive algorithm
2. Fast genus 1 theta-constants (Dupont, '06)
   - Theta-constants and AGM
   - A Newton-based algorithm
3. Fast genus 1 theta function (L., '15)
   - A quadratically convergent sequence
   - Quasi-optimal time algorithm
4. Fast genus g theta function (L.-Thomé, '16)
   - Reduction & naive algorithm
   - Quasi-optimal time algorithm

Outline

Jacobi's theta function

Definition (Jacobi's theta function)

$$\theta(z,\tau) = \sum_{n\in\mathbb{Z}} q^{n^2} w^{2n}$$

with $q = e^{i\pi\tau}, w = e^{i\pi z}$.

Definition (theta-constants)

$$\theta_0(0,\tau) = \sum_{n\in\mathbb{Z}} q^{n^2}, \quad \theta_1(0,\tau) = \sum_{n\in\mathbb{Z}} (-1)^n q^{n^2}, \quad \theta_2(0,\tau) = \sum_{n\in\mathbb{Z}} q^{n^2+n}$$

i.e. $\theta(0,\tau), \theta\left(\frac{1}{2},\tau\right), \theta\left(\frac{\tau}{2},\tau\right)$.

Problem: multiprecision computation of $\theta(z,\tau)$ + theta-constants with precision P.

Why?

Applications:

- Modular functions:

  Elliptic modular function j

$$j(\tau) = 54 \frac{(\boldsymbol{\theta_0(0,\tau)}^8 + \theta_1(0,\tau)^8 + \theta_2(0,\tau)^8)^3}{\boldsymbol{\theta_0(0,\tau)}^8 \theta_1(0,\tau)^8 \theta_2(0,\tau)^8}$$

Why?

Applications:

- Modular functions:

  Elliptic modular function j

  Dedekind η function

$$\eta(\tau) = \frac{\boldsymbol{\theta_0(0,\tau)}\theta_1(0,\tau)\theta_2(0,\tau)}{2}$$

Why?

Applications:

- Modular functions:

  Elliptic modular function j

  Dedekind η function

- Coefficients of the Weierstrass form of the analytic representation of an elliptic curve

  $\mathbb{C}/(\mathbb{Z}+\tau\mathbb{Z}) \simeq E(\mathbb{C})$ defined by $y^2 = 4x^3 - 60g_2x - 140g_3$;

$$g_2 = \frac{2}{3}\left(\boldsymbol{\theta_0(0,\tau)}^8 + \theta_1(0,\tau)^8 + \theta_2(0,\tau)^8\right)$$

$$g_3 = \frac{4}{27}\left(\boldsymbol{\theta_0(0,\tau)}^4 + \theta_1(0,\tau)^4\right)\left(\boldsymbol{\theta_0(0,\tau)}^4 + \theta_2(0,\tau)^4\right)\left(\theta_1(0,\tau)^4 - \theta_2(0,\tau)^4\right)$$

Why?

Applications:

- Modular functions:

  Elliptic modular function j

  Dedekind η function

- Coefficients of the Weierstrass form of the analytic representation of an elliptic curve

  $\mathbb{C}/(\mathbb{Z}+\tau\mathbb{Z}) \simeq E(\mathbb{C})$ defined by $y^2 = 4x^3 - 60g_2x - 140g_3$;

- Explicit isomorphism above: $z \mapsto (\wp(z,\tau), \wp'(z,\tau))$ with

$$\wp(z,\tau) = \frac{\pi^2(\theta_2(0,\tau)^4 - \theta_1(0,\tau)^4)}{3} - \pi^2\theta_1(0,\tau)^2\theta_2(0,\tau)^2 \frac{\boldsymbol{\theta_0(z,\tau)^2}}{e^{i\pi/2}\theta_2(z+1/2,\tau)^2}$$

Why?

Applications:

- Modular functions:

  Elliptic modular function j
  Dedekind η function

- Coefficients of the Weierstrass form of the analytic representation of an elliptic curve

  $\mathbb{C}/(\mathbb{Z}+\tau\mathbb{Z}) \simeq E(\mathbb{C})$ defined by $y^2 = 4x^3 - 60g_2x - 140g_3$;

- Explicit isomorphism above: $z \mapsto (\wp(z,\tau), \wp'(z,\tau))$
- Heat equation: for $x \in \mathbb{R}, t \in \mathbb{R}^+$, $\boldsymbol{\theta_0(x, it)}$ verifies

$$\frac{\partial}{\partial t}(\boldsymbol{\theta_0(x, it)}) = \frac{1}{4\pi}\frac{\partial^2}{\partial x^2}(\boldsymbol{\theta_0(x, it)})$$

Multiprecision

We work in multiprecision (P-bit numbers for any P; GMP, MPFR, MPC...) and in fixed-point arithmetic (= integer arithmetic).

Let $\mathcal{M}(P)$ = time it takes to compute $\left(\sum_{i=0}^{P-1} a_i 2^{-i}\right)\left(\sum_{i=0}^{P-1} b_i 2^{-i}\right)$.

Schoolbook: $\mathcal{M}(P) = O(P^2)$; Karatsuba: $\mathcal{M}(P) = O(P^{1.68})$;

Schönage-Strassen: $\mathcal{M}(P) = O(P \log P \log\log P)$;

Fürer (2007): $\mathcal{M}(P) = O(P \log P 2^{O(\log^* P)})$.

($\log^*(P) = n$ such that $\underbrace{\log\log \ldots \log}_{n \text{ times}} P \leq 1$)

Inversion, square root also costs $\boldsymbol{O(\mathcal{M}(P))}$ (using smart Newton).

$\log, \exp, \pi$: $\boldsymbol{O(\mathcal{M}(P)\log P)}$ algorithm (quasi-optimal; uses the AGM).

Naive algorithm

$$S_B(z,\tau) = 1 + \sum_{1 \leq n \leq B} q^{n^2}(w^{2n} + w^{-2n})$$

Theorem

For $\tau \in \mathcal{F}$ (i.e. $|\tau| \geq 1$, $|\operatorname{Re}(\tau)| \leq 1/2$) and z reduced:

$$|\boldsymbol{\theta_0(z,\tau)} - S_B(z,\tau)| \leq 3|q|^{(B-2)^2}$$

Similar thing for $\theta_1(z,\tau), \theta_2(z,\tau)$ and the theta-constants.

$\rightarrow$ If $B = O\left(\sqrt{\frac{P}{\operatorname{Im}(\tau)}}\right)$, S_B is an approx within 2^{-P}.

$\rightarrow$ $e^{i\pi\tau}$: $O(\mathcal{M}(P)\log P)$; each term $O(\mathcal{M}(P))$ (recurrence relations).

Total complexity: $O\left(\mathcal{M}(P)\sqrt{\frac{P}{\operatorname{Im}(\tau)}}\right)$ (better as $\operatorname{Im}(\tau) \nearrow$)

Outline

1. Genus 1 theta function
   - Applications
   - Naive algorithm
2. Fast genus 1 theta-constants (Dupont, '06)
   - Theta-constants and AGM
   - A Newton-based algorithm
3. Fast genus 1 theta function (L., '15)
   - A quadratically convergent sequence
   - Quasi-optimal time algorithm
4. Fast genus g theta function (L.-Thomé, '16)
   - Reduction & naive algorithm
   - Quasi-optimal time algorithm

Arithmetico-geometric mean

Definition

$$a_0, b_0 \in \mathbb{R}^+, \qquad a_{n+1} = \frac{a_n + b_n}{2}, \quad b_{n+1} = \sqrt{a_n b_n}.$$

Then $\mathrm{AGM}(a, b) = \lim_{n\to\infty} a_n$. It converges quadratically: $\sim \log P$ iterations to get P bits of prec; $\boldsymbol{O(\mathcal{M}(P)\log P)}$ bit ops.

Generalizing to $a, b \in \mathbb{C}$: at each step, two choices for the sign of the square root.

Definition

For (a_n, b_n) verifying the AGM equations, a value of b_n is a **good choice** of signs if

$$|a_n - b_n| \leq |a_n + b_n|$$

Optimal AGM sequence = good choice of sign at each step.

Complex AGM

Choice of signs is crucial:

Theorem

- *An AGM sequence with **infinitely many bad choices** of sign converges (at least linearly) to 0;*
- *An AGM sequence with **finitely many bad choices** (in particular, optimal AGM sequences) converges **quadratically** to a **non-zero** limit.*

Define $\mathrm{AGM}(a, b) = \lim_{n\to\infty} a_n$ where (a_n, b_n) is optimal:
well-defined homogeneous complex function.
Then computing P digits of $\mathrm{AGM}(a, b)$ costs $O(\mathcal{M}(P)\log P)$.

Computation of $\log(z)$, and thus of $\exp(z)$, in $O(\mathcal{M}(P)\log P)$.

AGM and theta-constants

Proposition

$$\begin{aligned} \boldsymbol{\theta_0(0,2\tau)^2} &= \frac{\boldsymbol{\theta_0(0,\tau)^2} + \theta_1(0,\tau)^2}{2} \\ \theta_1(0,2\tau)^2 &= \boldsymbol{\theta_0(0,\tau)}\theta_1(0,\tau) \end{aligned}$$

i.e. $(\boldsymbol{\theta_0(0,2^n\tau)^2}, \theta_1(0,2^n\tau)^2)$ *is an AGM sequence.*

Actually for $\tau \in \mathcal{F}$ it is an optimal AGM sequence, so

$$\mathrm{AGM}(\boldsymbol{\theta_0(0,\tau)^2}, \theta_1(0,\tau)^2) = 1$$

Let's rephrase that using the homogeneity:

$$\mathrm{AGM}\left(1, \frac{\theta_1(0,\tau)^2}{\boldsymbol{\theta_0(0,\tau)^2}}\right) = \frac{1}{\boldsymbol{\theta_0(0,\tau)^2}}$$

Extracting τ

$$\theta_2(0,\tau)^2 = -i\tau\theta_1\left(0, \frac{-1}{\tau}\right)^2, \quad \boldsymbol{\theta_0(0,\tau)^2 = -i\tau\theta_0\left(0, \frac{-1}{\tau}\right)^2}$$

Which means that

$$\mathrm{AGM}\left(1, \frac{\theta_2(0,\tau)^2}{\boldsymbol{\theta_0(0,\tau)^2}}\right) = \frac{i}{\tau\boldsymbol{\theta_0(0,\tau)^2}}$$

Jacobi's formula: $\boldsymbol{\theta_0(0,\tau)^4} = \theta_1(0,\tau)^4 + \theta_2(0,\tau)^4$, so

$$z \mapsto \frac{\mathrm{AGM}(1,z)}{\mathrm{AGM}\left(1, \sqrt{1-z^2}\right)} - i\tau$$

has $\frac{\theta_1(0,\tau)^2}{\boldsymbol{\theta_0(0,\tau)^2}}$ as zero.

Uniform algo to compute theta-constants

Use Newton's algorithm to compute $\frac{\theta_1(0,\tau)^2}{\theta_0(0,\tau)^2}$?

- Computing $\sqrt{1-z^2}$ can lose lots of precision if z is close to 1, which it is eventually; ($\sqrt{0.01} = 0.08?\ 0.12?$)
- Precision loss $= O(\mathrm{Im}(\tau))$: algo gets worse as $\mathrm{Im}(\tau)$ grows.

Naive algorithm to the rescue:

- If $c\,\mathrm{Im}(\tau) \geq P$, only need $\sqrt{c}$ terms in the series; naive algo costs $O(\mathcal{M}(P)\log P)$;
- If $c\,\mathrm{Im}(\tau) \leq P$, the algo above loses $O(P)$ digits $\rightarrow$ work at precision $2P$, doesn't change the asymptotics.

So there's an algo that computes theta-constants for any τ in $O(\mathcal{M}(P)\log P)$.

In genus 2

In genus 2: Borchardt mean of four numbers:

$$\mathcal{B}_2(x,y,z,t) = \left(\frac{x+y+z+t}{4}, \frac{\sqrt{x}\sqrt{y}+\sqrt{z}\sqrt{t}}{2}, \frac{\sqrt{x}\sqrt{z}+\sqrt{y}\sqrt{t}}{2}, \frac{\sqrt{x}\sqrt{t}+\sqrt{y}\sqrt{z}}{2}\right)$$

Good generalization of the AGM:

- *converges quadratically*;
- link with τ-duplication formula for genus 2 theta-constants

See Régis Dupont's PhD:

$O(\mathcal{M}(P)\log P)$ algorithm for genus 2 theta-constants;

Application: class polynomials in genus 2 (Enge-Thomé).

More on this later.

Outline

Strategy for theta function

Just like previously:

Find a quadratically convergent sequence involving theta-functions;

Find a function f that gives z, τ when evaluated on some values of theta-functions;

Use two-dimensional Newton method on $f : \mathbb{C}^2 \to \mathbb{C}^2$.

Our idea: use τ-duplication formulas for $\boldsymbol{\theta_0(z,\tau)}, \theta_1(z,\tau)$ to create something that looks like the AGM.

τ-duplication formulas

$$\boldsymbol{\theta_0(z,2\tau)^2} = \frac{\boldsymbol{\theta_0(z,\tau)\theta_0(0,\tau)} + \theta_1(z,\tau)\theta_1(0,\tau)}{2}$$
$$\theta_1(z,2\tau)^2 = \frac{\boldsymbol{\theta_0(z,\tau)}\theta_1(0,\tau) + \theta_1(z,\tau)\boldsymbol{\theta_0(0,\tau)}}{2}$$

hence define $M(x,y,z,t) = \left(\frac{\sqrt{x}\sqrt{z}+\sqrt{y}\sqrt{t}}{2}, \frac{\sqrt{x}\sqrt{t}+\sqrt{y}\sqrt{z}}{2}, \frac{z+t}{2}, \sqrt{z}\sqrt{t}\right)$.

We can define a good choice of roots, and for some values of z,τ:

- $M(\theta_0^2(z,\tau),\theta_1^2(z,\tau),\theta_0^2(0,\tau),\theta_1^2(0,\tau)) = (\theta_0^2(z,2\tau),\theta_1^2(z,2\tau),\theta_0^2(z,2\tau),\theta_1^2(0,2\tau)$
- $\lim_{n\to\infty} M^n(\boldsymbol{\theta_0(z,\tau)}^2,\theta_1(z,\tau)^2,\boldsymbol{\theta_0(0,\tau)}^2,\theta_1(0,\tau)^2) = (1,1,1,1)$

But it doesn't always converge quadratically:

$$M^n(2,2,1,1) = (2^{1/2^n},2^{1/2^n},1,1)$$

Homogenization

Be stubborn: let's try to homogenize, just like with the AGM:

$$\begin{aligned}(x'_n, y'_n, z'_n, t'_n) &= M^n(\lambda x, \lambda y, \mu z, \mu t)\\ &= \left(\lambda^{1/2^n}\mu^{1-1/2^n}x_n, \lambda^{1/2^n}\mu^{1-1/2^n}y_n, \mu z_n, \mu t_n\right)\end{aligned}$$

So

$$\mu = \frac{\lim_{n\to\infty} z'_\infty}{\lim_{n\to\infty} z_\infty}, \quad \lambda = \frac{\lim_{n\to\infty}\left(\frac{x'_n}{z'_\infty}\right)^{2^n} z'_\infty}{\lim_{n\to\infty}\left(\frac{x_n}{z_\infty}\right)^{2^n} z_\infty}$$

Proposition (L.)

$\left(\frac{x_n}{z_\infty}\right)^{2^n}$ converges quadratically; hence we can compute λ, μ in $O(\mathcal{M}(P)\log P)$.

The function we were looking for

$$\begin{aligned} \mathfrak{F} : \mathbb{C}^4 &\to \mathbb{C}^2 \\ (x, y, z, t) &\to \left(\left(\frac{x_n}{z_\infty} \right)^{2^n} \times z_\infty, z_\infty \right) \end{aligned}$$

where $(x_n, y_n, z_n, t_n) = M^n(x, y, z, t)$. We have

$$\mathfrak{F}\left(\lambda \boldsymbol{\theta_0}(z, \tau)^2, \lambda \theta_1(z, \tau)^2, \mu \boldsymbol{\theta_0}(0, \tau)^2, \mu \theta_1(0, \tau)^2\right) = (\lambda, \mu)$$

$$\mathfrak{F}\left(1, \frac{\theta_1(z, \tau)^2}{\boldsymbol{\theta_0}(z, \tau)^2}, 1, \frac{\theta_1(0, \tau)^2}{\boldsymbol{\theta_0}(0, \tau)^2}\right) = \left(\frac{1}{\boldsymbol{\theta_0}(z, \tau)^2}, \frac{1}{\boldsymbol{\theta_0}(0, \tau)^2}\right)$$

Action of $SL_2(\mathbb{Z})$

For theta constants, we looked at $\tau' = \frac{-1}{\tau}$. Similarly:

$$\theta_2(z,\tau)^2 = (-i\tau)e^{2i\pi z^2/\tau}\theta_1\left(\frac{z}{\tau}, \frac{-1}{\tau}\right)^2$$

$$\boldsymbol{\theta_0(z,\tau)^2 = (-i\tau)e^{2i\pi z^2/\tau}\theta_0\left(\frac{z}{\tau}, \frac{-1}{\tau}\right)^2}$$

$$\Rightarrow \quad \mathfrak{F}\left(1, \frac{\theta_2(z,\tau)^2}{\boldsymbol{\theta_0(z,\tau)^2}}, 1, \frac{\theta_2(0,\tau)^2}{\boldsymbol{\theta_0(0,\tau)^2}}\right) = \left(\frac{-i\tau e^{2i\pi z^2/\tau}}{\boldsymbol{\theta_0(z,\tau)^2}}, \frac{-i\tau}{\boldsymbol{\theta_0(0,\tau)^2}}\right)$$

Given $\frac{\theta_1(z,\tau)}{\boldsymbol{\theta_0(z,\tau)}}, \frac{\theta_1(0,\tau)}{\boldsymbol{\theta_0(0,\tau)}}$: apply $\mathfrak{F}$ then apply $\mathfrak{F}$ to $\frac{\theta_2(z,\tau)}{\boldsymbol{\theta_0(z,\tau)}}, \frac{\theta_2(0,\tau)}{\boldsymbol{\theta_0(0,\tau)}}$; recover z, τ.

Newton $\to O(\mathcal{M}(P)\log P)$.

Right choice of signs

The previous results only work for **some values of z, τ**:

Theorem (L.)

The right choice of square roots gives values of θ for

$$|\operatorname{Re}(\tau)| \leq \frac{1}{2}, \qquad \operatorname{Im}(\tau) > 0.345,$$

(for instance, $\tau \in \mathcal{F}$ or even $2\tau \in \mathcal{F}$)

$$|\operatorname{Re}(z)| \leq \frac{1}{8}, \qquad 0 \leq \operatorname{Im}(z) \leq \operatorname{Im}(\tau)/4$$

Once again, precision loss for θ_2 increases with $\operatorname{Im}(\tau)$.

$\Rightarrow$ Use τ-duplication formulas, z-duplication formulas to deduce result from computation in a compact.

Algorithm

- Assume $\tau \in \mathcal{F}$, $|\mathrm{Re}(z)| \leq \frac{1}{2}$ and $0 \leq \mathrm{Im}(z) < \mathrm{Im}(\tau)/2$.
- If $P \leq 25\,\mathrm{Im}\,\tau$:

  Use the naive algorithm.
- If not:
  - Pick $s \geq 1$ such that $\frac{1}{2} \leq |\tau/2^s| < 1$.
  - Compute all the $\theta\left(\frac{z}{2^{s+1}}, \frac{\tau}{2^s}\right)$ with precision P_0 **with the naive algorithm**.
  - Use $O(\log P/P_0)$ iterations of Newton's method to refine this into $\theta\left(\frac{z}{2^{s+1}}, \frac{\tau}{2^s}\right)$ with precision P.
  - Use a z-duplication formula, then (s times) use a τ-duplication formula then a z-duplication formula (to keep the size under control).

$\rightarrow$ uniform complexity of $O(\mathcal{M}(P)\log P)$ since $s = O(\log P)$.

Implementation

Implementation in C (MPC); compare timings (in s) to

- our MPC implem of the naive algorithm computing simultaneously $\boldsymbol{\theta_0(z,\tau)}, \boldsymbol{\theta_0(0,\tau)}, \theta_1(z,\tau), \theta_1(0,\tau)$;
- Magma's Theta function.

Prec (digits)	Magma	Naive	Fast
4000	0.17	0.03	0.09
16000	4.36	0.38	0.87
64000	116	4.60	6.78
256000	2426	41.6	45.5
325000		63.9	62.7
1024000		390	264
4096000		3921	1468

Outline

Generalization of Jacobi's theta

Definition

For $z \in \mathbb{C}^g$ and τ $g \times g$ matrix, sym. and $\mathrm{Im}(\tau) > 0$:

$$\theta(z,\tau) = \sum_{n \in \mathbb{Z}^g} e^{i\pi^t n \tau n} e^{2i\pi^t n z}$$

For $a, b \in \frac{1}{2}\mathbb{Z}^g/\mathbb{Z}^g$ $\quad$ $\theta_{[a;b]}(z,\tau) = \sum_{n \in \mathbb{Z}^g} e^{i\pi^t (n+a)\tau(n+a)} e^{2i\pi^t (n+a)(z+b)}$

Similar properties and formulas to genus 1.

Almost all the algorithm generalizes to genus g;

We show $g = 2$, which definitely works in $O(\mathcal{M}(P)\log P)$.

Background

Applications:

- In genus 2: isogeny computation (Robert & al.), fast arithmetic, class polynomial computations.
- In genus g: general Riemann surfaces (Mumford vol. 2).

Already done:

- Naive algorithm in genus g (Deconinck & al., 2004).
- Fast algorithm for theta constants in genus 2 (Dupont's PhD), applied to record computation of class polynomials (Enge & Thomé 2014).
- Hints of generalization to genus g.

Fundamental domain

In genus 1: $\mathcal{F} = \left\{|\tau| \geq 1, |\mathrm{Re}(\tau)| \leq \frac{1}{2}\right\}$.

fundamental domain for the action of $SL_2(\mathbb{Z})$ on $\mathcal{H}$

In genus g: similar action of $2g \times 2g$ symplectic matrices on $\mathcal{H}_g$.

Definition ($\mathcal{F}_g$, fundamental domain in genus g)

- $|\mathrm{Re}(\tau_{i,j})| \leq \frac{1}{2}$.
- $\mathrm{Im}(\tau)$ is Minkowski-reduced.
- For any $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ symplectic, $|\det(C\tau + D)| \geq 1$.

Fundamental domain

In genus 1: $\mathcal{F} = \left\{|\tau| \geq 1, |\mathrm{Re}(\tau)| \leq \frac{1}{2}\right\}$.

fundamental domain for the action of $SL_2(\mathbb{Z})$ on $\mathcal{H}$

In genus g: similar action of $2g \times 2g$ symplectic matrices on $\mathcal{H}_g$.

Definition ($\mathcal{F}_g$, fundamental domain in genus g)

- $|\mathrm{Re}(\tau_{i,j})| \leq \frac{1}{2}$. (easy)
- $\mathrm{Im}(\tau)$ is Minkowski-reduced. (hard : $O(2^{1.3g^3})$)
- For any $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ symplectic, $|\det(C\tau + D)| \geq 1$. (no known algorithm!)

Fundamental domain

In genus 1: $\mathcal{F} = \left\{|\tau| \geq 1, |\mathrm{Re}(\tau)| \leq \frac{1}{2}\right\}$.

fundamental domain for the action of $SL_2(\mathbb{Z})$ on $\mathcal{H}$

In genus g: similar action of $2g \times 2g$ symplectic matrices on $\mathcal{H}_g$.

Definition ($\mathcal{F}_g$, fundamental domain in genus g)

- $|\mathrm{Re}(\tau_{i,j})| \leq \frac{1}{2}$ (easy)
- $\mathrm{Im}(\tau)$ is Minkowski-reduced (hard : $O(2^{1.3g^3})$)
- For a finite number of $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ symplectic, $|\det(C\tau + D)| \geq 1$

In genus 2, Gottschling (1959) gave all 19 M;

In genus g, no description!

Weaker conditions

Definition (domain $\mathcal{B}_g$)

- $|\mathrm{Re}(\tau_{i,j})| \leq \frac{1}{2}$;
- $\mathrm{Im}(\tau)$ is Minkowski-reduced;
- $|\tau_{1,1}| \geq 1$.

Algorithm: Minkowski-reduce, if $|\tau_{1,1}| < 1$ apply some matrix N_0; rinse and repeat.

This terminates (Streng (2014) in genus 2; arguments are generalizable to genus g).

Number of steps: worse than g^g; each step costs $O(2^{1.3g^3})$.

Naive algorithm

Theorem (Deconinck & al. (2004))

Assuming "the exponential growth is factored out", it is enough to sum terms for which $|\sqrt{\pi}T(n+c)| < R$, *where* $\tau = {}^tTT$, *and with* R *defined as the solution of* $2^{-P} = \frac{g2^g}{2\rho^g}\Gamma(g/2, (R-\rho/2)^2)$.

i.e. an ellipsoid; we can prove that $R = O(\sqrt{P})$, so $O(P^{g/2})$ terms.

Assuming each term can be computed using induction relations: $O(\mathcal{M}(P)P^{g/2})$.

In genus 2:

More straightforward analysis: $O\left(\frac{P}{\mathrm{Im}(\tau_{11})}\right)$ terms.

Explicit induction relations between terms: $O\left(\mathcal{M}(P)\frac{P}{\mathrm{Im}(\tau_{11})}\right)$.

Implemented (MAGMA script); timings to follow.

τ-duplication formulas

$$\theta_{[a;b]}(z,\tau)^2 = \frac{1}{2^g} \sum_{\beta \in \frac{1}{2}\mathbb{Z}^g/\mathbb{Z}^g} e^{-4i\pi^t a\beta} \theta_{[0;b+\beta]}\left(z, \frac{\tau}{2}\right) \theta_{[0;\beta]}\left(0, \frac{\tau}{2}\right).$$

(i.e. $\theta_{0,\ldots,2^g-1}(0,\tau) \to$ all the $\theta_i(0,2\tau)^2$).

Define the function: $F(a_{0,\ldots,3}, b_{0,\ldots,3}) =$

$$\left(\frac{\sqrt{a_0}\sqrt{b_0} + \sqrt{a_1}\sqrt{b_1} + \sqrt{a_2}\sqrt{b_2} + \sqrt{a_3}\sqrt{b_3}}{4}, \frac{\sqrt{a_0}\sqrt{b_1} + \sqrt{a_1}\sqrt{b_0} + \sqrt{a_2}\sqrt{b_3} + \sqrt{a_3}\sqrt{b_2}}{4}, \right.$$
$$\frac{\sqrt{a_0}\sqrt{b_2} + \sqrt{a_1}\sqrt{b_3} + \sqrt{a_2}\sqrt{b_0} + \sqrt{a_3}\sqrt{b_1}}{4}, \frac{\sqrt{a_0}\sqrt{b_3} + \sqrt{a_1}\sqrt{b_2} + \sqrt{a_2}\sqrt{b_1} + \sqrt{a_3}\sqrt{b_0}}{4},$$
$$\left. \frac{b_0 + b_1 + b_2 + b_3}{4}, \frac{2\sqrt{b_0}\sqrt{b_1} + 2\sqrt{b_2}\sqrt{b_3}}{4}, \frac{2\sqrt{b_0}\sqrt{b_2} + 2\sqrt{b_1}\sqrt{b_3}}{4}, \frac{2\sqrt{b_0}\sqrt{b_3} + 2\sqrt{b_1}\sqrt{b_2}}{4} \right)$$

AGM-style

Conjecture

*The **good choices of sign** (the ones giving us quadratic convergence) are always the **choices of sign for θ** (the ones giving us θ) for $\tau \in \mathcal{B}_g$.*

- Eventually, the θ are close to 1, so it's true; is it true for all τ?
- Choice of sign is then easy (compute 2^g low-prec approx)

Genus 1 trick still works: $\lambda = \lim_{n\to\infty} \left(\frac{a_{0,n}}{b_{0,n}}\right)^{2^n} \times b_{0,n}$. Also

$$\lim_{n\to\infty} F^\infty\left(1, \frac{\theta_1^2, \theta_2^2, \theta_3^2(z,\tau)}{\theta_0^2(z,\tau)}, 1, \frac{\theta_1^2, \theta_2^2, \theta_3^2(0,\tau)}{\theta_0^2(0,\tau)}\right) = \left(\frac{1}{\theta_0(z,\tau)^2}, \frac{1}{\theta_0(0,\tau)^2}\right)$$

$\mathfrak{F} : \theta \mapsto (\lambda, \mu)$ still converges quadratically: $O(2^g \mathcal{M}(P) \log P)$.

Action of $Sp_4(\mathbb{Z})$

Theorem

$$F^\infty\left(\frac{\theta^2_{9,0,1}}{\theta^2_8}(z,\tau),\frac{\theta^2_{9,0,1}}{\theta^2_8}(0,\tau)\right) = \frac{e^{-2i\pi z_1^2/\tau_{11}}}{-\tau_{11}\theta_8(z,\tau)^2} = \frac{\lambda_1}{\mu_1\theta_8(z,\tau)^2}$$

$$F^\infty\left(\frac{\theta^2_{0,6,2}}{\theta^2_4}(z,\tau),\frac{\theta^2_{0,6,2}}{\theta^2_4}(0,\tau)\right) = \frac{e^{-2i\pi z_2^2/\tau_{22}}}{-\tau_{22}\theta_4(z,\tau)^2} = \frac{\lambda_2}{\mu_2\theta_4(z,\tau)^2}$$

$$F^\infty\left(\frac{\theta^2_{8,4,12}}{\theta^2_0}(z,\tau),\frac{\theta^2_{8,4,12}}{\theta^2_0}(0,\tau)\right) = \frac{e^{-2i\pi\frac{z_1^2\tau_{22}+z_2^2\tau_{11}-2z_1z_2\tau_{12}}{\det(\tau)}}}{(\tau_{12}^2-\tau_{11}\tau_{22})\theta_8(z,\tau)^2} = \frac{\lambda_3}{\mu_3\theta_8(z,\tau)^2}$$

How do you compute $\theta_{[a,b]}$ from $\theta_{[0,b]}$?

There are relations, but they are complicated (genus g?);

Use τ-duplication: get $\theta_{[a,b]}(z,2\tau)$ and a slightly different λ,τ.

Newton?

Maybe?
$$\begin{aligned} \mathfrak{F} : \mathbb{C}^6 &\to \mathbb{C}^5 \\ \frac{\theta_{1,2,3}(z,\tau)}{\theta_0(z,\tau)}, \frac{\theta_{1,2,3}(0,\tau)}{\theta_0(0,\tau)} &\mapsto (z_1, z_2, \tau_{11}, \tau_{22}, \tau_{12}) \end{aligned}$$

Not good for Newton! Two solutions:

Start with variety of dimension 5 = add Kummer surface.

End in space of dimension 6 = compute something more

Must keep the Jacobian invertible!

λ_3 works (why? because it's exp(complicated)?)

This works:
$$\begin{aligned} \mathfrak{F} : \mathbb{C}^6 &\to \mathbb{C}^6 \\ \frac{\theta_{1,2,3}(z,\tau)}{\theta_0(z,\tau)}, \frac{\theta_{1,2,3}(0,\tau)}{\theta_0(0,\tau)} &\mapsto (\lambda_1, \lambda_2, \lambda_3, \mu_1, \mu_2, \mu_3) \end{aligned}$$

(How do we do this in genus g?)

The algorithm

Similar to genus 1:

If $P \leq c\,\mathrm{Im}(\tau_{1,1})$, just use the naive algo;

If not:

- Divide z, τ by powers of two until in a chosen compact.
- Compute a first approximation of $\theta_{0,1,2,3}(z,\tau), \theta_{0,1,2,3}(0,\tau)$.
- Using Newton's method on $\mathfrak{F}$.

  Choice of signs: compute low-prec θ (thanks, compact).
- Use τ-duplication and z-duplication formulas.

Running time: $O(\mathcal{M}(P)\log P)$, uniformly.

Results

Timings in s:

Prec (digits)	Magma	Naive	This work
1000	0.42	0.38	0.38
2000	2.58	1.86	1.86
4000	18.4	9.51	6.65
8000	129	53.8	13.2
16000	889	303	25
32000	6368	1535	50
64000	46566	8798	120

Much better much earlier ($\log P$ versus P instead of vs $\sqrt{P}$)

Left for future work

Genus 1 & 2:

- Implementation of "batch thetas" ($\theta(z_i, \tau)$: cache theta-constant-related things)
  - Useful application: evaluating isogenies over $\mathbb{C}$.

Genus g:

- Prove things (quadratic convergence in genus g, good choice $= \theta$).
- Newton on $\mathfrak{F} : \mathbb{C}^{2\times(2^g-1)} \to \mathbb{C}^{g+\frac{g(g+1)}{2}}$??
  - Find the action of the right matrices so that the Jacobian is still invertible.
- Efficient implementation of the genus g naive algorithm?
  - (implementation of the fast algorithm is more straightforward)

That's all folks!

Thanks for your attention!