Evaluating theta functions in uniform quasi-linear time in any dimension

Jean Kieffer (CNRS, Loria)

NuSCAP annual meeting, Paris, May 23, 2024

Joint work with Noam D. Elkies

Plan of the talk

1. Introduction: evaluating theta functions
2. The “naive” algorithm
3. The duplication formula
4. The final algorithm

The Riemann theta function

Parameters:

- $g \geq 1$: the dimension (sometimes called genus)
- $\tau \in \mathcal{H}_g$, the Siegel upper half-space: this means $\tau \in \mathrm{Mat}_{g\times g}(\mathbb{C})$ is symmetric and $\mathrm{Im}(\tau)$ is positive definite ($y^T \mathrm{Im}(\tau) y > 0$ for all nonzero $y \in \mathbb{R}^g$).
- $z \in \mathbb{C}^g$.

Define the Riemann theta function:

$$\theta(z,\tau) = \sum_{n\in\mathbb{Z}^g} E(n^T\tau n + 2n^T z).$$

where $E(x) := \exp(\pi i x)$. This sum converges quickly (terms get small as $n \to \infty$).

If $g = 1$, this is the Jacobi theta function

$$\theta(z,\tau) = \sum_{n\in\mathbb{Z}} E(\tau n^2 + 2nz).$$

Theta functions with characteristics

More generally, for all theta characteristics $a, b \in \{0,1\}^g$, define:

$$\theta_{a,b}(z,\tau) = \sum_{n \in \mathbb{Z}^g + \frac{a}{2}} E\left(n^T \tau n + 2n^T (z + \tfrac{b}{2})\right).$$

Before, we had $a = b = 0$.

Remark

Up to an exponential factor, $\theta_{a,b}(z,\tau)$ is simply $\theta_{0,0}(z + \tau\frac{a}{2} + \frac{b}{2}, \tau)$. The reason for this notation will become clear on the next slide.

Why theta functions?

Theta functions are closely connected to elliptic curves and abelian varieties over $\mathbb{C}$.

1. If τ is fixed and z varies, then $\theta(\cdot, \tau)$ is (roughly) periodic with respect to the lattice $L = \mathbb{Z}^g + \tau\mathbb{Z}^g \subset \mathbb{C}^g$.

   The quotient $A = \mathbb{C}^g/L$ is an abelian variety of dimension g, and theta functions with characteristics are coordinate functions on A. For instance, take A to be the Jacobian of any algebraic curve.
2. Fixing $z = 0$ and letting τ vary, the theta constants $\theta_{a,b}(0, \cdot)$ are modular forms. They can be used as invariants to identify an abelian variety, a curve, etc.

These properties (periodicity, modular forms) are also generally helpful when manipulating theta functions, as we will see.

Evaluating theta functions

Algorithmic problem

Given $(z, \tau) \in \mathbb{C}^g \times \mathcal{H}_g$, and a precision $N \geq 0$, compute the complex numbers $\theta_{a,b}(z, \tau)$ for all $a, b \in \{0, 1\}^g$ at precision N up to an error of at most 2^{-N}.

In applications, N can be in the millions.

Theorem (in progress, joint with Noam D. Elkies)

There exists an algorithm which, given $g \geq 1$, $N \geq 0$, and given (z, τ) that are suitably reduced, evaluates $\theta_{a,b}(z, \tau)$ to precision N in quasi-linear time $O(2^{O(g \log g)} \mathcal{M}(N) \log N)$ uniformly in τ and z.

Based on the duplication formula. Implemented in FLINT 3.1.

Brief history of previous work

- The naive algorithm (see Deconinck et al., 2002) consists in summing up enough terms in the theta series

$$\theta_{a,b}(z,\tau) = \sum_{n\in\mathbb{Z}^g + \frac{a}{2}} E\big(n^T \tau n + 2n^T(z + \tfrac{b}{2})\big).$$

  Useful at low precisions, but not quasi-linear. Optimized in the $g = 1$ case by Enge–Hart–Johansson (2018).
- Dupont (2006), Labrande–Thomé (2010): quasi-linear algorithm based on a clever use of Newton's method. Heuristic, mainly tested for $g \leq 3$. Does not beat the naive algorithm for $g = 1$ in the feasible range.
- In some cases ($g \leq 2$) one can prove that the Newton approach works and yields a uniform algorithm (K., 2022). Still not known to work for all τ as soon as $g \geq 3$.

Brief list of available implementations

Implementations based on the naive algorithm:

- `Theta.jl` by Agostini–Chua (2020), low precision only.
- Magma's `Theta`, arbitrary g and precisions, extremely slow.
- `RiemannTheta`, Sage package by Nils Bruin, arbitrary g and precisions, less slow.
- `acb_modular` (FLINT) by Enge–Hart–Johansson (2018). $g = 1$ only, uses interval arithmetic, fast.

Often also support theta functions with characteristics, derivatives.

Implementations based on Newton's method exist, but are not easily accessible.

New implementation: `acb_theta` in FLINT 3.1. Any g, fast, quasi-linear, uniform, uses interval arithmetic, supports characteristics and derivatives, extensively tested.

Use that one!

The “naive” algorithm

Convergence of the theta series

Recall:

$$\theta_{0,0}(z,\tau) = \sum_{n\in\mathbb{Z}^g} E(n^T\tau n + 2n^T z).$$

Write $Y = \mathrm{Im}(\tau)$ and $y = \mathrm{Im}(z)$. Then:

$$\begin{aligned} \left|E(n^T\tau n + 2n^T z)\right| &= \exp(-\pi n^T Y n - 2\pi n^T y) \\ &= C\exp(-\|n - x_0\|_\tau^2) \end{aligned}$$

where $\|\cdot\|_\tau$ is the Euclidean norm attached to πY, and C, x_0 depend only on τ and z.

Useful consequence

For each $N \geq 0$, the lattice points $n \in \mathbb{Z}^g$ indexing terms whose absolute value is $\geq 2^{-N}$ are exactly the points in an ellipsoid $\|n - x_0\|_\tau \leq R$ with $R = O(\sqrt{N})$.

The tail of the series

Proposition

Let $\tau \in \mathcal{H}_g$, let C be the Cholesky matrix attached to $\pi\,\mathrm{Im}(\tau)$ (meaning: C is upper-triangular and $\pi\,\mathrm{Im}(\tau) = C^T C$), and let $\gamma_1, \ldots, \gamma_g$ be the diagonal coefficients of C. For each $R \geq 4$, we have:

$$\sum_{n \in \mathbb{Z}^g,\, \|n - x_0\|_\tau > R} \exp(-\|n - x_0\|_\tau^2) \leq 2^{g+1} R^{g-1} \exp(-R^2) \prod_{i=1}^{g} \Big(1 + \frac{2}{\gamma_i}\Big).$$

$\to$ taking $R = O(\sqrt{N})$, the tail of the series is bounded by 2^{-N}.

A first step when evaluating is to obtain nicer ellipsoids (i.e. larger γ_i) by reducing the arguments τ and z.

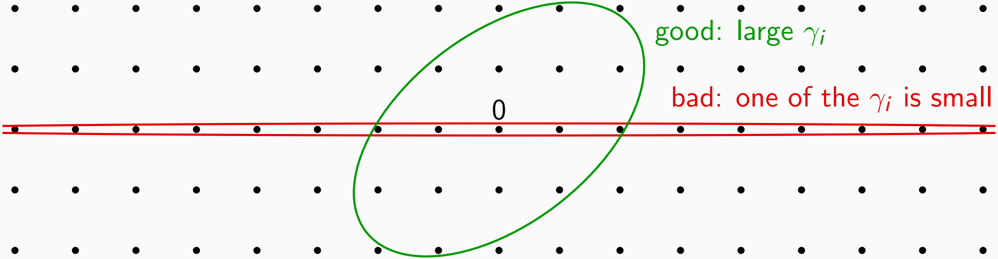
Argument reduction

Easy reductions using periodicity:

- reduce τ such that $|\mathrm{Re}(\tau)| \leq \frac{1}{2}$ (changes nothing)
- reduce z such that $z = u + \tau v$ with $u, v \in \mathbb{R}^g$ and $|u|, |v| \leq \frac{1}{2}$

  $\rightarrow$ the center of the ellipsoid is not far away from 0.

We also want to reduce $\mathrm{Im}(\tau)$. This changes the shape of the ellipsoids.



The symplectic group

The symplectic group $\mathrm{Sp}_{2g}(\mathbb{Z})$ consists of products of matrices of the form

$$\begin{pmatrix} I_g & S \\ 0 & I_g \end{pmatrix}, \qquad S \in \mathrm{Mat}_{g\times g}(\mathbb{Z}) \text{ symmetric}$$

$$\begin{pmatrix} U & 0 \\ 0 & U^{-T} \end{pmatrix}, \qquad U \in \mathrm{GL}_g(\mathbb{Z})$$

$$J_g := \begin{pmatrix} 0 & I_g \\ -I_g & 0 \end{pmatrix}.$$

Equivalently, all matrices M such that $M^T J_g M = J_g$.

The group $\mathrm{Sp}_{2g}(\mathbb{Z})$ acts on $\mathbb{C}^g \times \mathcal{H}_g$:

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \cdot (z,\tau) = \Big((\gamma\tau+\delta)^{-T} z,\ (\alpha\tau+\beta)(\gamma\tau+\delta)^{-1} \Big).$$

Theta as a modular form

Theorem (Mumford, Igusa '60s)

Let $M = \left(\begin{smallmatrix}\alpha & \beta \\ \gamma & \delta\end{smallmatrix}\right) \in \mathrm{Sp}_{2g}(\mathbb{Z})$. Fix theta characteristics a, b. For every $(z, \tau) \in \mathbb{C}^g \times \mathcal{H}_g$:

$$\theta_{a,b}(M \cdot (z, \tau)) = \exp(\cdots) \cdot \zeta \cdot \sqrt{\det(\gamma\tau + \delta)} \cdot \theta_{a',b'}(z, \tau)$$

where:

- a', b' are other characteristics given by an explicit formula in terms of a, b, M,
- ζ is an 8th root of unity independent of z, τ,
- we must make a fixed choice of holomorphic sqrt of $\tau \mapsto \det(\gamma\tau + \delta)$ on $\mathcal{H}_g$.

Sort out the details $\to$ can act by $\mathrm{Sp}_{2g}(\mathbb{Z})$ before computing theta functions.

A nicer imaginary part

We have

$$\mathrm{Im}\big((\alpha\tau + \beta)(\gamma\tau + \delta)^{-1}\big) = (\gamma\tau + \delta)^{-T}\,\mathrm{Im}(\tau)(\gamma\tau + \delta)^{-1}.$$

Therefore:

- Can use lattice reduction (LLL) by taking $M = \begin{pmatrix} U & 0 \\ 0 & U^{-T} \end{pmatrix}$.
- Can increase $\det \mathrm{Im}(\tau)$ whenever $\left|\det(\gamma\tau + \delta)\right| < 1$. In particular, use the usual reduction for the action of $\mathrm{SL}_2(\mathbb{Z})$ on $\mathcal{H}_1$ on each diagonal coefficient.

Consequence

We can assume that $\mathrm{Im}(\tau)$ is LLL-reduced and its diagonal coefficients are $\geq \sqrt{3}/2$.

The ellipsoids we get in the naive algorithm are uniformly nice and round.

Naive algorithm: complexity

Result

Given reduced $(z, \tau) \in \mathbb{C}^g \times \mathcal{H}_g$ and $N \geq 1$, one can compute $\theta_{a,b}(z, \tau)$ for each individual characteristic (a, b) to N bits of precision using the naive algorithm in $O_g(N^{g/2}\mathcal{M}(N))$ binary operations, uniformly in τ and z.

Important optimizations in practice (but not really new):

- Compute exponentials only once, get subsequent terms by multiplying/squaring.
- Compute smaller terms (far from the center of the ellipsoid) at smaller precisions.
- Use existing functions when possible: `acb_modular_theta` ($g = 1$), `acb_dot`.

In FLINT, we implement ellipsoids as a recursive type to make all this easier to write.

Towards a quasi-linear algorithm

The naive algorithm is not quasi-linear... except when $\mathrm{Im}(\tau)$ is large!

If τ is reduced and each diagonal coefficient of $\mathrm{Im}(\tau)$ is $\Omega(N)$, then the ellipsoid to sum on contains $O(1)$ points $\rightarrow$ complexity $O_g(\mathcal{M}(N)\log N)$.

The duplication formula relates theta values at τ and 2τ.

Main idea

Use the duplication formula $k \simeq \log_2 N$ times starting from theta values at $2^k\tau$, computed in quasi-linear time with the naive algorithm.

The duplication formula

The typical duplication formula

The duplication formula is also used in the Newton approach to computing theta functions (Dupont '06, Labrande–Thomé '16).

We identify $\{0,1\}^g$ and $(\mathbb{Z}/2\mathbb{Z})^g$ (addition is xor).

Duplication formula

$$\theta_{a,b}(0,2\tau)^2 = \frac{1}{2^g} \sum_{b' \in (\mathbb{Z}/2\mathbb{Z})^g} (-1)^{a^T b'} \theta_{0,b'}(0,\tau)\, \theta_{0,b+b'}(0,\tau).$$

Note the particular role of $\theta_{0,b}$ compared to more general $\theta_{a,b}$.

For us, this goes in the wrong direction: we want to express theta values at τ in terms of theta values at 2τ.

A better formula

Cf. Koizumi, or Romain Cosset's thesis, or apply J_g to the previous formula:

Better duplication formula

$$\theta_{a,b}(0,\tau)^2 = \sum_{a' \in (\mathbb{Z}/2\mathbb{Z})^g} (-1)^{a'^T b} \theta_{a',0}(0,2\tau)\, \theta_{a+a',0}(0,2\tau).$$

This time, the 2^g "fundamental" theta values are the $\theta_{a,0}(0,\tau)$. For now, we focus on computing those and put $z = 0, b = 0$.

To apply the duplication formula, we need to:

1. Make the 2^g sums on the right (one for each $a \in \{0,1\}^g$, with $b = 0$). Do this in $O(2^g)$ operations with Hadamard transformations.
2. Extract square roots: get $\theta_{a,b}(0,\tau)$ from $\theta_{a,0}(0,\tau)^2$.

The root problem

Problems when extracting square roots:

1. We need to know what the correct sign is $\rightarrow$ need a low-precision approximation of $\theta_{a,0}(0,\tau)$. Can compute it using the naive algorithm.
2. Taking square roots brings a precision loss, perhaps as much as half of the current precision since $\sqrt{2^{-N}} = 2^{-N/2}$.

Both problems get worse the closer $\theta_{a,0}(0,\tau)$ gets to zero.

Dream Scenario

There exists $\varepsilon > 0$ such that for all $k \geq 0$ and $a \in \{0,1\}^g$, we have $\left|\theta_{a,0}(0, 2^k\tau)\right| \geq \varepsilon$.

This is however just false, since $\theta_{a,0}(0, 2^k\tau) \underset{k\to\infty}{\longrightarrow} 0$ as soon as $a \neq 0$.

Need to quantify this to show that we're not killed by the naive algorithm and/or precision losses.

The absolute value of theta

Recall our previous analysis: the term corresponding to $n \in \mathbb{Z}^g + \frac{a}{2}$ in the series defining $\theta_{a,0}(0,\tau)$ has absolute value $\exp(-\|n\|_\tau^2)$.

Dream Scenario 2

There exists $\varepsilon > 0$ such that for each $k \geq 0$, we have

$$\left|\theta_{a,0}(0, 2^k\tau)\right| \geq \varepsilon \exp\left(-2^k \operatorname{Dist}_\tau(0, \mathbb{Z}^g + \tfrac{a}{2})\right)$$

Here Dist_τ denotes the distance (between point and set) attached to the norm $\|\cdot\|_\tau$.

In other words $|\theta_{a,0}(\tau)|$ is comparable to the absolute value of the largest term appearing in the sum – no crazy cancellation occurs.

We expect this to be true (with a reasonable ε) for almost every τ.

The dream world

Assume Dream Scenario 2. Then each time we apply the duplication formula:

- Computing an approximation of $\theta_{a,0}(0,\tau)$ with the naive algorithm costs $O(1)$.
- We lose $O(1)$ bits of precision in square roots, provided that we think in terms of shifted absolute precision.

Convention

By "computing $\theta_{a,0}(0, 2^k\tau)$ to shifted absolute precision N", we mean computing it to absolute precision $N + \left\lceil 2^k \operatorname{Dist}_\tau(0, \mathbb{Z}^g + \frac{a}{2}) / \log(2) \right\rceil$.

This accounts for the fact that $\left|\theta_{a,0}(0, 2^k\tau)\right|$ is known to be small. In Dream Scenario 2, this is the same as relative precision.

- Small miracle: when summing in the duplication formula, we also lose only $O(1)$ bits of shifted absolute precision (parallelogram identity!)
- To initialize at $2^k = O(N)$, we use the naive algorithm and win.

The final algorithm

The real world

- For some special τ's, we might have unexpected vanishings of $\theta_{a,0}(0, 2^k\tau)$. Then the previous algorithm does not work.
- We also want to compute $\theta_{a,0}(z, \tau)$ for nonzero z.

Observation

Let $t \in \mathbb{R}^g$ be any vector. If, at each step, we compute $\theta_{a,0}(2^k v, 2^k\tau)$ for all $a \in \{0,1\}^g$ and all $v \in \{0, t, 2t, z, z+t, z+2t\}$, then we can bootstrap using variants of the duplication formula.

This requires us to take square roots of $\theta_{a,0}(2^k v, 2^k\tau)^2$ for $v \in \{t, 2t, z+t, z+2t\}$, but not $v = 0$ and $v = z$ (get those by division).

Introducing the real vector t changes nothing to ellipsoids and distances, but can prevent unexpected cancellations. In practice, a random t does the trick.

Theoretical result

Proposition (writeup in progress)

Fix $g \geq 1$ and $m \geq 0$. Then there exists $\varepsilon > 0$ such that for a proportion at least $1/2$ of vectors $t \in [0,1]^g$, the following holds:

For each reduced $(z,\tau) \in \mathbb{C}^g \times \mathcal{H}_g$, for each $a \in \{0,1\}^g$, for each $0 \leq k \leq m$, and for each $v \in \{t, 2t\}$, we have

$$\left|\theta_{a,0}(2^k v, 2^k \tau)\right| \geq \varepsilon \exp\left(-2^k \operatorname{Dist}_\tau(0, \mathbb{Z}^g + \tfrac{a}{2})\right),$$
$$\left|\theta_{a,0}(2^k (z+v), 2^k \tau)\right| \geq \varepsilon \exp\left(-2^k \operatorname{Dist}_\tau(x_0, \mathbb{Z}^g + \tfrac{a}{2})\right)$$

where x_0 denotes the center of the ellipsoid attached to z and $\varepsilon = m^{-\mathrm{Poly}(g)}$.

Choosing t at random, precision losses are mild with a probability $\geq 1/2$. ✓

Further comments

- If one of the diagonal coefficients γ_i is very large, the ellipsoids for $\|\cdot\|_\tau$ are thick in some directions and very thin in other directions. We leverage this by writing $\theta_{a,0}(z,\tau)$ as a (short) sum of theta values for smaller g.
- This algorithm overcomes FLINT's implementation of the naive algorithm for $g = 1$ between 10 000 and 50 000 bits of precision. I'm sure this can be improved.
- We also compute derivatives of theta functions in quasi-linear time using finite differences with rigorous error bounds.

Thank you!

https://flintlib.org/doc/acb_theta.html