

Fast evaluation of Riemann theta functions in any dimension

Jean Kieffer

Joint work with Noam D. Elkies

FoCM workshop on Computational Number Theory, 13 July 2026



Riemann theta functions

For an integer $g \geq 1$, define the Siegel upper half-space

$$\mathcal{H}_g = \{ \tau \in \text{Mat}_{g \times g}(\mathbb{C}) \text{ s.t. } \tau \text{ is symmetric and } \text{Im}(\tau) \text{ is positive definite} \}.$$

Throughout, $\tau \in \mathcal{H}_g$ and $z \in \mathbb{C}^g$.

Riemann theta functions

For an integer $g \geq 1$, define the **Siegel upper half-space**

$$\mathcal{H}_g = \{ \tau \in \text{Mat}_{g \times g}(\mathbb{C}) \text{ s.t. } \tau \text{ is symmetric and } \text{Im}(\tau) \text{ is positive definite} \}.$$

Throughout, $\tau \in \mathcal{H}_g$ and $z \in \mathbb{C}^g$.

The **Riemann theta function** is defined by the convergent series

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

Riemann theta functions

For an integer $g \geq 1$, define the **Siegel upper half-space**

$$\mathcal{H}_g = \{ \tau \in \text{Mat}_{g \times g}(\mathbb{C}) \text{ s.t. } \tau \text{ is symmetric and } \text{Im}(\tau) \text{ is positive definite} \}.$$

Throughout, $\tau \in \mathcal{H}_g$ and $z \in \mathbb{C}^g$.

The **Riemann theta function** is defined by the convergent series

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

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

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

Riemann theta functions

For an integer $g \geq 1$, define the **Siegel upper half-space**

$$\mathcal{H}_g = \{ \tau \in \text{Mat}_{g \times g}(\mathbb{C}) \text{ s.t. } \tau \text{ is symmetric and } \text{Im}(\tau) \text{ is positive definite} \}.$$

Throughout, $\tau \in \mathcal{H}_g$ and $z \in \mathbb{C}^g$.

The **Riemann theta function** is defined by the convergent series

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

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

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

We care as number theorists because there is a close link between theta functions and **abelian varieties over $\mathbb{C}$** .

Abelian varieties over $\mathbb{C}$

- **Algebraic varieties**: any abelian variety A embeds as a subvariety of $\mathbb{P}_{\mathbb{C}}^n$ for some n .
- Compact **complex tori**: $A = \mathbb{C}^g / \Lambda$ where $g = \dim(A)$ and Λ is a lattice of rank $2g$.
If A is principally polarized, we can take $\Lambda = (\tau\mathbb{Z}^g + \mathbb{Z}^g)$ for some $\tau \in \mathcal{H}_g$.
Such a τ is called a **period matrix** of A .
- **Group varieties**: A carries an algebraic addition law inherited from $+$ on $\mathbb{C}^g / \Lambda$.

Theta functions relates the analytic world of complex tori to algebraic geometry.

Theta functions and abelian varieties

For a fixed $\tau \in \mathcal{H}_g$, let $A_\tau = \mathbb{C}^g / (\tau\mathbb{Z}^g + \mathbb{Z}^g)$. Consider the map

$$\begin{aligned}\Theta_\tau : \mathbb{C}^g &\rightarrow \mathbb{C}^{2^g} \\ z &\mapsto (\theta_{a,b}(z, \tau))_{a,b}.\end{aligned}$$

The **periodicity relations** satisfied by θ ensure that Θ_τ induces a map

$$\Theta_\tau : A_\tau \rightarrow \mathbb{P}_{\mathbb{C}}^n, \quad \text{where } n = 2^g - 1.$$

Fundamental theorems (Riemann, Mumford)

- Θ_τ is a projective embedding of A_τ .
- The **theta-null point** $\Theta_\tau(0) \in \mathbb{P}_{\mathbb{C}}^n$ determines a set of equations for $\Theta_\tau(A_\tau)$, hence determines A_τ up to isomorphism.
- Given $\Theta_\tau(0)$, $\Theta_\tau(z)$ and $\Theta_\tau(z')$, there are simple formulas to compute $\Theta_\tau(z + z')$.

Numerical evaluation

To use the previous theorems in **algorithms**, we face the following problem:

Evaluate Riemann theta functions

Input: $g \geq 1$, $\tau \in \mathcal{H}_g$, $z \in \mathbb{C}^g$, and a **precision** N , measured in bits.

Output: $\theta_{a,b}(z, \tau)$ as complex numbers, up to an error of at most 2^{-N} , for all characteristics (a, b) .

In practice g is small (there are 2^{2g} values) but N can be large, up to millions of bits.

Applications include:

- Computing models/invariants of abelian varieties from a period matrix,
- Computing isogenies, modular polynomials, class polynomials,
- Solving the (explicit) inverse Galois problem, etc.

For the remainder of this talk, we focus on the evaluation problem.

Evaluation algorithms

Three algorithms

- The **summation algorithm(s)**: compute a partial sum of the defining series. The state of the art for $g \geq 3$ prior to this work.
- Our new **fast algorithm** based on duplication formulas.

In between, **Dupont's algorithm** (2006) is based on the AGM and Newton iterations. Fast, but restricted to $g \leq 2$ or maybe 3, and hard to make rigorous.

Argument reduction

Before applying any of those, one has to **reduce** (z, τ) using the periodicity relations satisfied by $\theta_{a,b}$ to ensure that the theta series converges rapidly. In particular $\text{Im}(\tau)$ will be LLL- or HKZ-reduced, and its diagonal coefficients are bounded below.

The summation algorithm

Let's look more closely at the convergence of the theta series

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

The summation algorithm

Let's look more closely at the convergence of the theta series

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

Let $y = \text{Im}(z)$. For a given n , we have

$$|\exp(\pi i(n^T \tau n + 2n^T z))| = \exp(\pi y^T \text{Im}(\tau)^{-1} y) \cdot \exp(-\|n - v\|_\tau^2)$$

where $\|\cdot\|_\tau$ is the Euclidean norm attached to $\pi \text{Im}(\tau)$, and $v = -\text{Im}(\tau)^{-1} y \in \mathbb{R}^g$.

The summation algorithm

Let's look more closely at the convergence of the theta series

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

Let $y = \text{Im}(z)$. For a given n , we have

$$|\exp(\pi i(n^T \tau n + 2n^T z))| = \exp(\pi y^T \text{Im}(\tau)^{-1} y) \cdot \exp(-\|n - v\|_\tau^2)$$

where $\|\cdot\|_\tau$ is the Euclidean norm attached to $\pi \text{Im}(\tau)$, and $v = -\text{Im}(\tau)^{-1} y \in \mathbb{R}^g$.

Useful consequence

For a given N , the indices $n \in \mathbb{Z}^g$ of terms in the series whose absolute values are $\geq 2^{-N}$ are the **integer points in an ellipsoid** $\|n - v\|_\tau \leq R$ with $R = O(\sqrt{N})$.

In fact, if we increase R slightly, the partial sum over that ellipsoid will be a valid approximation of $\theta_{0,0}(z, \tau)$ up to an error of 2^{-N} .

In the interest of getting uniform complexity bounds even for large z , define

$$\tilde{\theta}_{a,b}(z, \tau) = \exp(-\pi y^T \operatorname{Im}(\tau)^{-1} y) \theta_{a,b}(z, \tau)$$

Theorem (Deconinck et al., 2004, $+\varepsilon$)

Given $z \in \mathbb{C}^g$, $\tau \in \mathcal{H}_g$ and a precision N , the summation algorithm evaluates $\tilde{\theta}_{a,b}(z, \tau)$ up to an error of 2^{-N} in time $2^{O(g \log g)} N^{1+g/2} \log N$.

This complexity bound is uniform in (z, τ) if reduced.

Proof sketch: the ellipsoid contains $O(N^{g/2})$ points.

In the interest of getting uniform complexity bounds even for large z , define

$$\tilde{\theta}_{a,b}(z, \tau) = \exp(-\pi y^T \operatorname{Im}(\tau)^{-1} y) \theta_{a,b}(z, \tau)$$

Theorem (Deconinck et al., 2004, $+\varepsilon$)

Given $z \in \mathbb{C}^g$, $\tau \in \mathcal{H}_g$ and a precision N , the summation algorithm evaluates $\tilde{\theta}_{a,b}(z, \tau)$ up to an error of 2^{-N} in time $2^{O(g \log g)} N^{1+g/2} \log N$.

This complexity bound is uniform in (z, τ) if reduced.

Proof sketch: the ellipsoid contains $O(N^{g/2})$ points.

Main theorem (Elkies, K., 2026+)

One can evaluate $\tilde{\theta}_{a,b}(z, \tau)$ up to an error of 2^{-N} in time $2^{O(g \log^2 g)} N (\log N)^2$. This complexity bound is uniform in (z, τ) if reduced.

Fast evaluation: outline

The fast algorithm

Relies on two observations:

- The summation algorithm is **fast** if τ is “high up” in $\mathcal{H}_g$. If τ is reduced and each diagonal coefficient of $\text{Im}(\tau)$ is $\Omega(N)$, then the ellipsoid contains $O(1)$ points.
- The **duplication formula** relates theta values at τ and 2τ :

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

Rough outline

Use the duplication formula $h \simeq \log_2 N$ times. Initialize using theta values at $2^h \tau$, computed using the summation algorithm.

But there are still some issues.

Extracting square roots

As a special case of the duplication formula ($x = x' = 0$), we have

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

So we have to extract square roots to continue. To **decide the correct sign**, we compute a low-precision value by summation.

Extracting square roots

As a special case of the duplication formula ($x = x' = 0$), we have

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

So we have to extract square roots to continue. To **decide the correct sign**, we compute a low-precision value by summation.

Key issue

If there are cancellations in the series and $\theta_{a,b}(0, \tau)$ is too close to zero, then:

- Determining the correct sign requires high precision, which would be too costly;
- Even if we knew the sign, we would face a deadly precision loss when extracting the square root, since $d(\sqrt{x})/dx \rightarrow +\infty$ as $x \rightarrow 0^+$.

Avoiding cancellations

If the key issue arises, we introduce an auxiliary “random” vector $t \in \mathbb{R}^g$:

- We modify the algorithm to only extract square roots of theta values involving t .
- We’ll get the values that are potentially too small by division instead (again by something that’s far enough from zero).

For instance we require that for all $0 \leq n \leq h - 1$,

$$|\theta_{a,0}(2^n t, 2^n \tau)| >$$

Avoiding cancellations

If the key issue arises, we introduce an **auxiliary “random” vector** $t \in \mathbb{R}^g$:

- We modify the algorithm to only extract square roots of theta values **involving** t .
- We'll get the values that are potentially too small by **division** instead (again by something that's far enough from zero).

For instance we require that for all $0 \leq n \leq h - 1$,

$$|\theta_{a,0}(2^n t, 2^n \tau)| > \exp(-2^n \text{Dist}(0, \mathbb{Z}^g + \frac{a}{2})^2)$$

where Dist is the distance associated with $\|\cdot\|_\tau$

Avoiding cancellations

If the key issue arises, we introduce an auxiliary “random” vector $t \in \mathbb{R}^g$:

- We modify the algorithm to only extract square roots of theta values involving t .
- We’ll get the values that are potentially too small by division instead (again by something that’s far enough from zero).

For instance we require that for all $0 \leq n \leq h - 1$,

$$|\theta_{a,0}(2^n t, 2^n \tau)| > \eta_g \exp(-2^n \text{Dist}(0, \mathbb{Z}^g + \frac{a}{2})^2)$$

where Dist is the distance associated with $\|\cdot\|_\tau$ and $\eta_g > 0$.

A possible workflow

$$\theta_{a,0}(0, 2\tau)$$

$$\theta_{a,0}(2t, 2\tau)$$

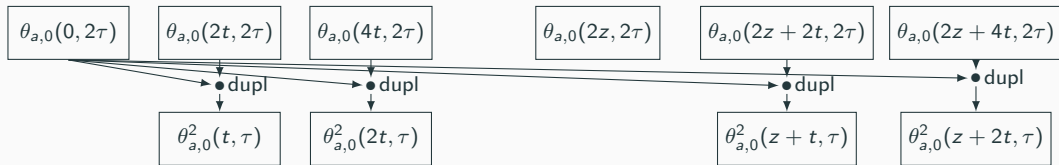
$$\theta_{a,0}(4t, 2\tau)$$

$$\theta_{a,0}(2z, 2\tau)$$

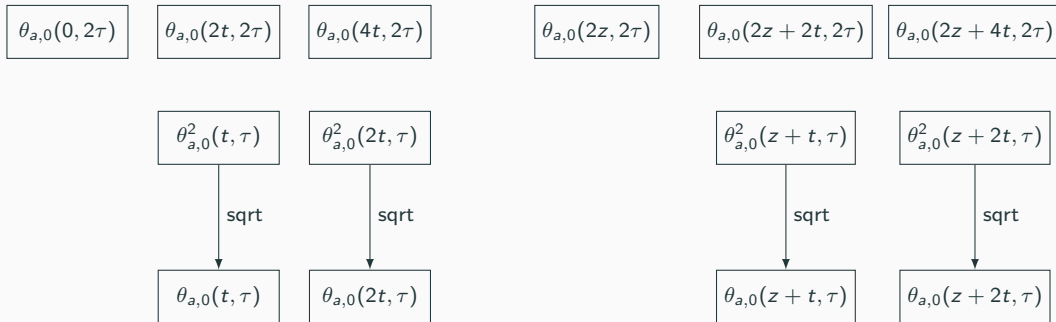
$$\theta_{a,0}(2z + 2t, 2\tau)$$

$$\theta_{a,0}(2z + 4t, 2\tau)$$

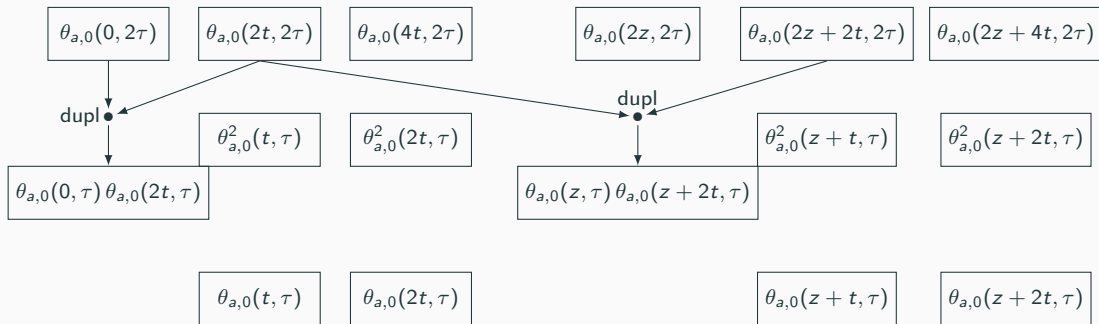
A possible workflow



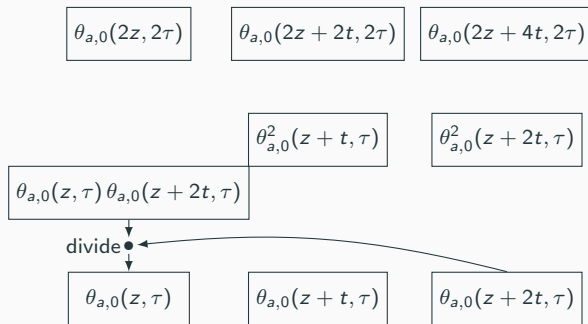
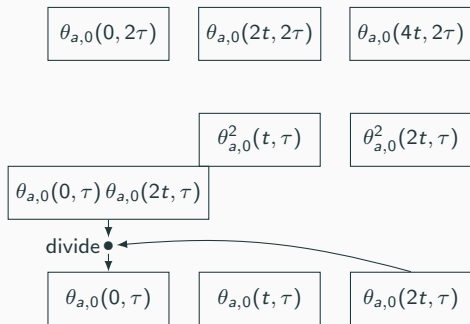
A possible workflow



A possible workflow



A possible workflow



Results in the paper

- We can deterministically choose t so that the required inequalities hold. The constant η_g depends explicitly on g .
- This suffices to control precision losses throughout, provided we use **shifted absolute precisions** $N + 2^n \text{Dist}(v, \mathbb{Z}^g + \frac{a}{2})^2 / \log(2)$. These shifted precisions are compatible with the duplication formula thanks to the parallelogram identity.
- If these distances get too large compared to N , this means τ is unbalanced. In that case we use **recursive calls in lower dimension** and fewer duplication steps.

Implementation

Real and complex numbers in FLINT

FLINT is a performant C library for number theory featuring interval arithmetic on real and complex numbers (formerly **Arb**, by F. Johansson).

A typical **arb_t** (arbitrary-precision real ball) consists of:

- an exact dyadic number ρ , the **midpoint**, of arbitrary precision.
- an exact dyadic **error radius** $\varepsilon \geq 0$.

It stands for any real number x such that $|x - \rho| \leq \varepsilon$.

An **acb_t** encodes a complex number; its real and imaginary parts are **arb_t**'s.

There are dedicated C types for vectors, polynomials, matrices, etc.

Operations are **mathematically meaningful**: error bounds are propagated automatically.

Theta functions in FLINT

Documentation (FLINT v3.3) is available at:

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



The main function to compute a single value $\theta_{a,b}(z, \tau)$ is

```
void acb_theta_one(acb_t th, acb_srcptr z, const acb_mat_t tau,
                  ulong ab, slong prec)
```

The “mothership” is `acb_theta_jet`, which can also evaluate

- partial derivatives (to any order),
- all characteristics at the same time,
- and/or squared values $\theta_{a,b}^2$ (more efficiently).

For the summation algorithm (no reduction included!), use `acb_theta_sum`.

Performance comparison: summation

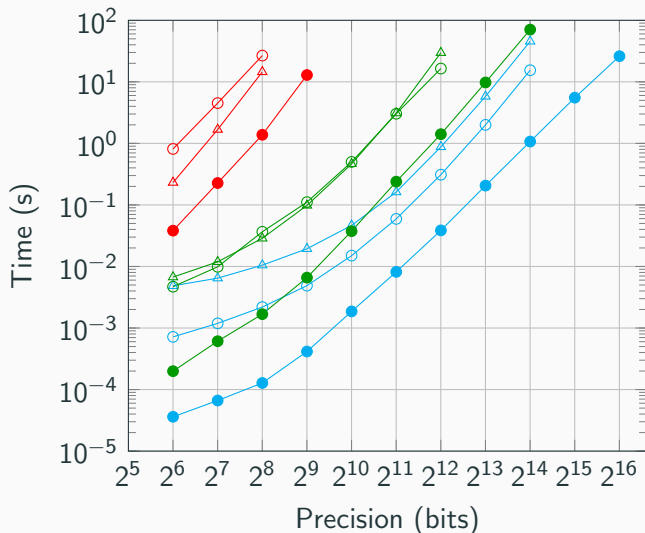
$\theta_{0,0}(\tau)$ for a typical τ

Dimensions 2, 3, 6

acb_theta: ●

Magma: ○

RiemannTheta (Sage): △

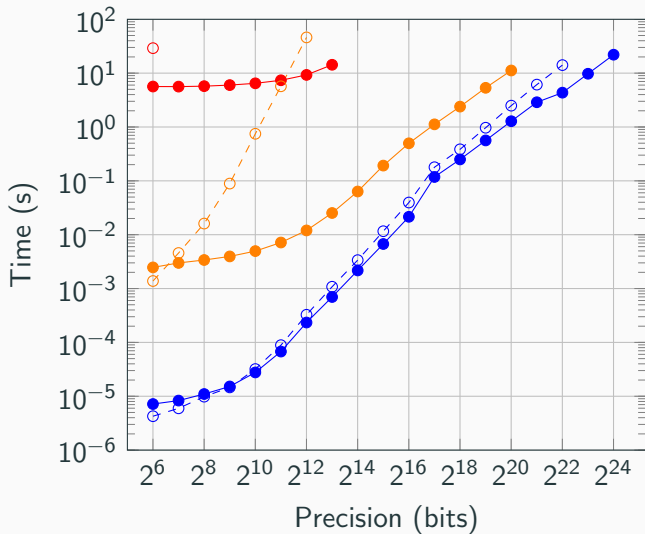


Performance comparison in FLINT: fast algorithm vs. summation

$\theta_{0,0}(\tau)$ for a typical τ

Dimensions 1, 4, 10

Dashed line is summation, solid line is the fast algorithm.



- Future work to improve both the FLINT code and theoretical bounds, especially in the high genus/low precision range.
- Could the fast evaluation of θ help for the fast computation of period matrices, reversing Dupont's approach?

Thank you!