Computing the truncated theta function via Mordell integral

A. Kuznetsov *

Dept. of Mathematics and Statistics
York University
4700 Keele Street
Toronto, ON
M3J 1P3, Canada

Current version: March 12, 2014

Abstract

Hiary [3] has presented an algorithm which allows to evaluate the truncated theta function $\sum_{k=0}^{n} \exp(2\pi \mathrm{i}(zk+\tau k^2))$ to within $\pm\epsilon$ in $O(\ln(\frac{n}{\epsilon})^{\kappa})$ arithmetic operations for any real z and τ. This remarkable result has many applications in Number Theory, in particular it is the crucial element in Hiary's algorithm for computing $\zeta(\frac{1}{2}+it)$ to within $\pm t^{-\lambda}$ in $O_{\lambda}(t^{\frac{1}{3}}\ln(t)^{\kappa})$ arithmetic operations, see [2]. We present a significant simplification of Hiary's algorithm for evaluating the truncated theta function. Our method avoids the use of the Poisson summation formula, and substitutes it with an explicit identity involving the Mordell integral. This results in an algorithm which is efficient, conceptually simple and easy to implement.

Keywords: truncated theta function, Mordell integral, Riemann zeta function
2010 Mathematics Subject Classification : 11Y16, 11M06

*Research supported by the Natural Sciences and Engineering Research Council of Canada. The author would like to thank the anonymous referee for careful reading of the paper and for providing many valuable comments and suggestions. The first version of this paper was written while the author was visiting the Department of Mathematical Sciences of the University of Bath, whose hospitality is highly appreciated.

1 Introduction and main results

The truncated theta function is defined as

$$F_n(z,\tau) := \sum_{k=0}^{n} e^{2\pi \mathrm{i}(zk+\tau k^2)}, \tag{1}$$

where $n \in \mathbb{N}$ and $z, \tau \in \mathbb{R}$. We are interested in computing efficiently the truncated theta function $F_n(z,\tau)$ for large values of n. The motivation for developing such algorithms comes from computational Number Theory. One particularly important application is for computing the Riemann zeta function $\zeta(\frac{1}{2}+\mathrm{i}t)$ for large t. The classical result by Riemann and Siegel (see formula 4.17.5 in [11]) allows to evaluate $\zeta(\frac{1}{2}+\mathrm{i}t)$ to within $\pm t^{-\lambda}$ in $O_\lambda(t^{\frac{1}{2}})$ arithmetic operations (by an arithmetic operation we mean additions, multiplications, evaluations of the logarithm and of the complex exponential function). It is a very challenging task to prove that the number of operations can be reduced to $O(t^\alpha)$ with some $\alpha < 1/2$. It is an even harder problem to develop an algorithm which would not only have theoretical complexity $O(t^\alpha)$, but would also be feasible for practical implementation. In the recent papers [2, 3], Hiary has developed two new algorithms, which allow us to evaluate $\zeta(\frac{1}{2}+\mathrm{i}t)$ to within $\pm t^{-\lambda}$ in $O_\lambda(t^{\frac{1}{3}}\ln(t)^\kappa)$ and $O_\lambda(t^{\frac{4}{13}}\ln(t)^\kappa)$) arithmetic operations (we will refer to them as 1/3-algorithm and 4/13-algorithm). The 1/3-algorithm is particularly attractive compared with the 4/13-algorithm, as it is conceptually simpler, has no substantial storage requirements and should be easier to implement. And this brings us back to the truncated theta functions, as the 1/3-algorithm is fundamentally based on another result by Hiary (see [3, Theorem 1.1]), which states that $F_n(z,\tau)$ and its derivatives can be computed to within $\pm\epsilon$ in $O(\ln\left(\frac{n}{\epsilon}\right)^\kappa)$ arithmetic operations.

Let us summarize the main ideas behind Hiary's truncated theta function algorithm. The algorithm is based on the iterative application of the following two-step procedure. The first step is to "normalize" z and τ so that they lie in the intervals $z \in [-1/2, 1/2]$ and $\tau \in [-1/4, 1/4]$. This is easily achieved via the identities

$$F_n(z,\tau) = F_n\left(z+\tfrac{1}{2}, \tau+\tfrac{1}{2}\right) = F_n(z+k, \tau+l), \quad k,l \in \mathbb{Z}, \tag{2}$$

both of which follow directly from (1). If $|\tau| < 1/n$, then we compute $F_n(z,\tau)$ by the Euler-Maclaurin summation (see Section 3.2 in [3]), otherwise we proceed to the second step of the algorithm, which is based on the following identity

$$F_n(z,\tau) = \frac{e^{\frac{\pi\mathrm{i}}{4} - \frac{\pi\mathrm{i}z^2}{2\tau}}}{\sqrt{2\tau}} F_m\left(\tfrac{z}{2\tau}, -\tfrac{1}{4\tau}\right) + R_{m,n}(z,\tau), \tag{3}$$

where $\tau > 0$ and $m = \lfloor 2n\tau \rfloor$. Here and everywhere in this paper $\lfloor x \rfloor := \max\{k \in \mathbb{Z} \ : \ k \le x\}$ denotes the floor function. A corresponding identity for $\tau < 0$ can be derived by taking the conjugate of both sides of (3) and using the fact that $\overline{F_n(z,\tau)} = F_n(-z,-\tau)$.

Identity (3) plays the central role in Hiary's algorithm. It is derived by applying the Poisson summation formula to (1) and using the crucial self-similarity property of the Gaussian function (the fact that taking Fourier transform of $\exp(-ax^2-bx-c)$ results in a function of the same form but with different parameters). The truncated theta function $F_m(\cdot,\cdot)$ in the right-hand side of (3) is the dominant contribution arising from the Poisson summation formula, while $R_{m,n}(z,\tau)$ can be considered as the remainder term. The key idea behind the second step in Hiary's algorithm is that the identity (3) transforms a long sum with n terms into a short sum having $m = \lfloor 2n|\tau| \rfloor \le n/2$ terms (recall that we have normalized

τ so that $|\tau| \leq 1/4$). If m is smaller than a fixed power of $\ln(n/\epsilon)$, then we compute $F_m(\cdot,\cdot)$ by direct summation, otherwise we apply the same two-step procedure to $F_m(\cdot,\cdot)$. It is clear that this algorithm will terminate after at most $\log_2(n)$ iterations (where $\log_2(\cdot)$ denotes the logarithm with base two).

Our goal in this paper is to simplify Hiary's algorithm for computing the truncated theta function. In order to present our results, first we need to introduce *Mordell integral*, which is defined as

$$h(z,\tau) := \int_{\mathbb{R}} \frac{e^{\pi \mathrm{i}\tau x^2 - 2\pi z x}}{\cosh(\pi x)} \mathrm{d}x, \quad z \in \mathbb{C},\ \mathrm{Im}(\tau) > 0. \tag{4}$$

We need to extend this definition for real values of τ. Assume that τ lies in the quadrant $\mathrm{Re}(\tau) > 0$ and $\mathrm{Im}(\tau) > 0$. Then the following identity is true

$$h(z,\tau) = e^{\frac{\pi \mathrm{i}}{4}} \int_{\mathbb{R}} \frac{e^{-\pi\tau y^2 - 2\pi z e^{\frac{\pi \mathrm{i}}{4}} y}}{\cosh(\pi e^{\frac{\pi \mathrm{i}}{4}} y)} \mathrm{d}y = 2e^{\frac{\pi \mathrm{i}}{4}} \int_0^{\infty} e^{-\pi\tau y^2} \frac{\cosh(2\pi z e^{\frac{\pi \mathrm{i}}{4}} y)}{\cosh(\pi e^{\frac{\pi \mathrm{i}}{4}} y)} \mathrm{d}y. \tag{5}$$

This result can be easily established by rotating the contour of integration $\mathbb{R}^+ \mapsto e^{\frac{\pi \mathrm{i}}{4}}\mathbb{R}^+$ in the integral (4) and then changing the variable of integration $x = e^{\frac{\pi \mathrm{i}}{4}} y$. It is clear the integral in (5) provides the analytic continuation of $h(z,\tau)$ into the domain $\mathrm{Re}(\tau) > 0$, $z \in \mathbb{C}$. We will adopt this as the definition of $h(z,\tau)$ for $\tau > 0$ and $z \in \mathbb{R}$, while for $\tau < 0$ and $z \in \mathbb{R}$ we will define $h(z,-\tau) := \overline{h(z,\tau)}$.

Mordell integral and related functions have been studied for more than a century. One particular example of such integrals was used by Riemann to derive the functional equation for the zeta function (see section 2.10 in [11]), while some further special cases were investigated by Ramanujan [9]. Mordell [6, 7] has developed a general theory of $h(z,\tau)$ and some related integrals. Among many other results, Mordell has discovered the connections between $h(z,\tau)$ and theta functions, he has studied general modular transformations of $h(z,\tau)$ and has proved that $h(z,\tau)$ can be expressed in terms of the Gauss sums when τ is rational (more precisely, $h(z,p/q)$ can be expressed in terms of functions $F_q(z,p/(2q))$ and $F_p(z,-q/(2p))$, see section 8 in [7]). Recently, the Mordell integral $h(z,\tau)$ was investigated by Zwegers in his Ph.D. thesis [13], and we will follow Zwegers's notation in our paper.

The following theorem is our first main result, and it is the basis for our simplified version of Hiary's algorithm for fast evaluations of $F_n(z,\tau)$.

Theorem 1. *For $\tau > 0$, $z \in \mathbb{R}$ and $m, n \in \mathbb{N} \cup \{0\}$*

$$F_n(z,\tau) = \frac{e^{\frac{\pi \mathrm{i}}{4} - \frac{\pi \mathrm{i} z^2}{2\tau}}}{\sqrt{2\tau}} F_m\left(\tfrac{z}{2\tau}, -\tfrac{1}{4\tau}\right) + R_{m,n}(z,\tau), \tag{6}$$

where

$$\begin{aligned} R_{m,n}(z,\tau) &= -\frac{\mathrm{i}}{2} e^{-\pi \mathrm{i}\left(z - \frac{\tau}{2}\right)} h\left(z - \tau + \tfrac{1}{2}, -2\tau\right) \\ &- \frac{\mathrm{i}}{2}(-1)^m e^{2\pi \mathrm{i}\left(n+\frac{1}{2}\right)\left(z+\tau\left(n+\frac{1}{2}\right)\right)} h\left(z + (2n+1)\tau - m - \tfrac{1}{2}, -2\tau\right). \end{aligned} \tag{7}$$

Using Theorem 1 we are able to establish the following result (which should be compared with Theorem 1.1 in [3]).

Theorem 2. *There exists an algorithm such that for any $\epsilon \in (0, \frac{1}{10})$, any $z, \tau \in (0,1)$ and any integer $n \geq 1$ the value of the function $F_n(z,\tau)$ can be evaluated to within $\pm\epsilon$ using $\leq C_1 \ln\left(\frac{n}{\epsilon}\right)^3$ arithmetic operations on numbers of $\leq C_2 \ln\left(\frac{n}{\epsilon}\right)$ bits. The algorithm requires $\leq C_3 \ln\left(\frac{n}{\epsilon}\right)^2$ bits of memory.*

In the above result (and everywhere in this paper) we assume that $A_1, A_2, \dots$ and $C_1, C_2, \dots$ are constants, which are positive and absolute, in the sense that they do not depend on the values of any other parameters.

The paper is organized as follows: Section 2 contains the proofs of Theorems 1 and 2 and in Section 3 we discuss the practical implementation and some extensions of the algorithm.

2 Proof of the main results

Lemma 1. *For $z \in \mathbb{R}$ and $\tau > 0$*

$$h(z,\tau) + h(z+1,\tau) = \frac{2}{\sqrt{\tau}} e^{\frac{\pi \mathrm{i}}{4} + \frac{\pi \mathrm{i}}{\tau}(z+\frac{1}{2})^2}, \tag{8}$$

$$h(z,\tau) = \frac{1}{\sqrt{\tau}} e^{\frac{\pi \mathrm{i}}{4} + \frac{\pi \mathrm{i} z^2}{\tau}} h(\tfrac{z}{\tau}, -\tfrac{1}{\tau}). \tag{9}$$

Proof. While the above identities are not new (see [7] and [13, Proposition 1.2]), we present the sketch of the proof for the sake of completeness. Let us denote $\theta := e^{\frac{\pi \mathrm{i}}{4}}$ and assume that $\tau > 0$. From the second integral representation in (5) we obtain

$$h(z) + h(z+1) = \theta \int_{\mathbb{R}} \frac{e^{-\pi\tau y^2 - 2\pi z\theta y}}{\cosh(\pi\theta y)} \left(1 + e^{-2\pi\theta y}\right) \mathrm{d}y = 2\theta \int_{\mathbb{R}} e^{-\pi\tau y^2 - 2\pi\theta y\left(z+\frac{1}{2}\right)} \mathrm{d}y. \tag{10}$$

Evaluating the integral in the right-hand side of the above identity (use formula (3.323.2) in [1]) gives us (8).

In order to prove (9) we use formulas (3.323.2) and (3.511.4) in [1] and evaluate the following two integrals: For $w \in \mathbb{R}$

$$\int_{\mathbb{R}} e^{-\pi\tau y^2 - 2\pi z\theta y - 2\pi \mathrm{i} y w} \mathrm{d}y = \frac{1}{\sqrt{\tau}} e^{\frac{\pi \mathrm{i}}{\tau}(z+\theta w)^2}, \qquad \theta \int_{\mathbb{R}} \frac{e^{-2\pi \mathrm{i} y w}}{\cosh(\pi\theta y)} \mathrm{d}y = \frac{1}{\cosh(\pi\bar{\theta} w)}. \tag{11}$$

Applying Parseval's Theorem for Fourier transform to the first integral in (5) gives us

$$h(z,\tau) = \frac{1}{\sqrt{\tau}} \int_{\mathbb{R}} \frac{e^{\frac{\pi \mathrm{i}}{\tau}(z+\theta w)^2}}{\cosh(\pi\bar{\theta} w)} \mathrm{d}w = \frac{1}{\sqrt{\tau}} e^{\frac{\pi \mathrm{i}}{4} + \frac{\pi \mathrm{i} z^2}{\tau}} \overline{h(z, \tfrac{1}{\tau})} = \frac{1}{\sqrt{\tau}} e^{\frac{\pi \mathrm{i}}{4} + \frac{\pi \mathrm{i} z^2}{\tau}} h(\tfrac{z}{\tau}, -\tfrac{1}{\tau}). \tag{12}$$

□

Proof of Theorem 1: We take the conjugate of both sides of equation (8) and obtain

$$h\left(z - \tfrac{1}{2}, -\tau\right) = -h\left(z + \tfrac{1}{2}, -\tau\right) + \frac{2}{\sqrt{\tau}} e^{-\frac{\pi \mathrm{i}}{4} - \frac{\pi \mathrm{i} z^2}{\tau}}. \tag{13}$$

Iterating the above identity m times gives us

$$h\left(z - \tfrac{1}{2}, -\tau\right) = (-1)^{m+1} h\left(z + m + \tfrac{1}{2}, -\tau\right) + \frac{2}{\sqrt{\tau}} \sum_{k=0}^{m} (-1)^k e^{-\frac{\pi \mathrm{i}}{4} - \frac{\pi \mathrm{i}}{\tau}(z+k)^2}, \tag{14}$$

which is equivalent to

$$h\left(z - \tfrac{1}{2}, -\tau\right) = (-1)^{m+1} h\left(z + m + \tfrac{1}{2}, -\tau\right) + \frac{2}{\sqrt{\tau}} e^{-\frac{\pi \mathrm{i}}{4} - \frac{\pi \mathrm{i} z^2}{\tau}} F_m\left(\tfrac{1}{2} - \tfrac{z}{\tau}, -\tfrac{1}{2\tau}\right). \tag{15}$$

We change variables $z = w/t$, $\tau = -1/t$ and $m = n$ in (15), and then apply transformation (9) to both h-functions, which results in the following identity

$$\begin{aligned} e^{\frac{\pi \mathrm{i}}{4}+\frac{\pi \mathrm{i}}{t}\left(w-\frac{t}{2}\right)^2} h\left(w-\tfrac{t}{2},-t\right) &= (-1)^{n+1} e^{\frac{\pi \mathrm{i}}{4}+\frac{\pi \mathrm{i}}{t}\left(w+nt+\frac{t}{2}\right)^2} h\left(w+nt+\tfrac{t}{2},-t\right) \\ &+ 2e^{\frac{\pi \mathrm{i}}{4}+\frac{\pi \mathrm{i} w^2}{t}} F_n\left(w-\tfrac{1}{2},\tfrac{t}{2}\right). \end{aligned} \tag{16}$$

At the same time, changing variables $z = w + nt - m + t/2 - 1/2$ and $\tau = t$ in (15) we obtain

$$\begin{aligned} h\left(w+nt-m+\tfrac{t}{2}-1,-t\right) &= (-1)^{m+1} h\left(w+nt+\tfrac{t}{2},-t\right) \\ &+ \frac{2}{\sqrt{t}} e^{-\frac{\pi \mathrm{i}}{4}-\frac{\pi \mathrm{i}}{t}\left(w+nt-m+\frac{t}{2}-\frac{1}{2}\right)^2} F_m\left(\tfrac{1}{t}\left(m+\tfrac{1}{2}-w\right),-\tfrac{1}{2t}\right). \end{aligned} \tag{17}$$

Eliminating $h\left(w+nt+\frac{t}{2},-t\right)$ from the two equations (16) and (17) gives us

$$\begin{aligned} & e^{\frac{\pi \mathrm{i}}{4}+\frac{\pi \mathrm{i}}{t}\left(w-\frac{t}{2}\right)^2} h\left(w-\tfrac{t}{2},-t\right) = 2e^{\frac{\pi \mathrm{i}}{4}+\frac{\pi \mathrm{i} w^2}{t}} F_n\left(w-\tfrac{1}{2},\tfrac{t}{2}\right) + (-1)^{n+1} e^{\frac{\pi \mathrm{i}}{4}+\frac{\pi \mathrm{i}}{t}\left(w+nt+\frac{t}{2}\right)^2} \\ \times\ & (-1)^{m+1}\left[h\left(w+nt-m+\tfrac{t}{2}-1,-t\right) - \frac{2}{\sqrt{t}} e^{-\frac{\pi \mathrm{i}}{4}-\frac{\pi \mathrm{i}}{t}\left(w+nt-m+\frac{t}{2}-\frac{1}{2}\right)^2} F_m\left(\tfrac{1}{t}\left(m+\tfrac{1}{2}-w\right),-\tfrac{1}{2t}\right)\right]. \end{aligned} \tag{18}$$

The above identity is equivalent to (6). In order to verify this one would need to apply the transformation

$$F_m\left(\tfrac{1}{t}\left(m+\tfrac{1}{2}-w\right),-\tfrac{1}{2t}\right) = e^{\frac{\pi \mathrm{i}}{t}(m^2-m(2w-1))} F_m\left(\tfrac{1}{t}\left(w-\tfrac{1}{2}\right),-\tfrac{1}{2t}\right), \tag{19}$$

(which follows easily from (1) by changing the index of summation $k \mapsto m-k$), then change the variables $w = z + \frac{1}{2}$ and $t = 2\tau$ and simplify the result. □

Now we need to introduce several new objects. We define the sequence $\{\tilde{E}_k\}_{k\geq 0}$ as

$$\frac{1}{\cosh(x)} = \sum_{k\geq 0} \tilde{E}_k x^k, \tag{20}$$

or, alternatively, $\tilde{E}_k = E_k/k!$ where $\{E_k\}_{k\geq 0}$ are Euler numbers. We define the function $f_\tau(z)$ and the sequence of polynomials $\{q_k(\tau)\}_{k\geq 0}$ as

$$f_\tau(z) := \frac{e^{\frac{\mathrm{i}\tau z^2}{\pi}}}{\cosh(z)} = \sum_{k\geq 0} q_k(\tau) z^{2k}. \tag{21}$$

Using equation (20) it is easy to see that

$$q_k(\tau) = \sum_{j=0}^{k} \tilde{E}_{2k-2j} \frac{1}{j!} \left(\frac{\mathrm{i}\tau}{\pi}\right)^j. \tag{22}$$

We introduce the sequence of functions $\{p_k(x)\}_{k\geq 0}$, defined by

$$p_k(x) := \int_0^1 e^{-xu} u^k \mathrm{d}u. \tag{23}$$

For $k \geq 1$, $\tau > 0$ and $z \in \mathbb{R}$ we define

$$H_k(z,\tau) := \frac{e^{\frac{\pi \mathrm{i}}{4}}}{\sqrt{\tau}} \sum_{l=0}^{k-1} (-1)^l e^{\frac{\pi \mathrm{i}}{\tau}(z+l+\frac{1}{2})^2} \left[1 - \Phi\left(\sqrt{\tfrac{\pi}{\tau}} e^{\frac{\pi \mathrm{i}}{4}} (z+l+\tfrac{1}{2})\right)\right], \tag{24}$$

where $\Phi(z)$ is the error function

$$\Phi(z) := \frac{2}{\sqrt{\pi}} \int_0^z e^{-x^2} \mathrm{d}x = \frac{2}{\sqrt{\pi}} \sum_{n \geq 0} \frac{(-1)^n}{n!} \frac{z^{2n+1}}{2n+1}, \quad z \in \mathbb{C}. \tag{25}$$

Finally, for $z > 0$ and $\tau \in \mathbb{R}$ we define

$$J(z,\tau) := \int_0^\infty e^{-2xz} f_\tau(x) \mathrm{d}x, \tag{26}$$

where $f_\tau(x)$ is given by (21).

Proposition 1.

(i) For $k \geq 1$, $|z| \leq 1/2$ and $\tau \in (0,1)$

$$h(z,\tau) = H_k(z,\tau) + H_k(-z,\tau) + \frac{(-1)^k}{\pi} \left(J(k+z,\tau) + J(k-z,\tau)\right). \tag{27}$$

(ii) For $\epsilon > 0$ let us denote $K = K(\epsilon) = 2 + 2\lceil \ln(\frac{1}{\epsilon}) \rceil$. Then for any $\epsilon \in (0, \frac{1}{10})$, $|z| \leq 1/2$ and $\tau \in (0,1)$

$$h(z,\tau) = H_K(z,\tau) + H_K(-z,\tau) + \frac{1}{\pi} \sum_{0 \leq l \leq K} q_l(\tau) \big(p_{2l}(2(K+z)) + p_{2l}(2(K-z))\big) + \mathcal{E}, \tag{28}$$

where $|\mathcal{E}| < \epsilon$.

Proof. Let us prove part (i). We denote $\theta := e^{\frac{\pi \mathrm{i}}{4}}$, use the second integral representation in (5) and the identity

$$\cosh(\pi\theta y)^{-1} = 2 \sum_{l=0}^{k-1} (-1)^l e^{-(2l+1)\pi\theta y} + (-1)^k e^{-2\pi k\theta y} \cosh(\pi\theta y)^{-1}, \tag{29}$$

and obtain

$$h(z,\tau) = 4\theta \sum_{l=0}^{k-1} (-1)^l \int_0^\infty e^{-\pi\tau y^2 - (2l+1)\pi\theta y} \cosh(2\pi z\theta y) \mathrm{d}y + 2\theta(-1)^k \int_0^\infty e^{-\pi\tau y^2 - 2\pi k\theta y} \frac{\cosh(2\pi z\theta y)}{\cosh(\pi\theta y)} \mathrm{d}y. \tag{30}$$

The integrals in the sum in the right-hand side of (30) can be evaluated explicitly by applying formula (3.322.2) in [1]; this gives us the terms $H_k(z,\tau) + H_k(-z,\tau)$. To deal with the remaining integral in the right-hand side of (30), we rotate the contour of integration $\mathbb{R}^+ \mapsto e^{-\frac{\pi \mathrm{i}}{4}} \mathbb{R}^+$ and change the variable of integration $y = x/(\pi\theta)$. This gives us

$$2\theta \int_0^\infty e^{-\pi\tau y^2 - 2\pi k\theta y} \frac{\cosh(2\pi z\theta y)}{\cosh(\pi\theta y)} \mathrm{d}y = \frac{1}{\pi} \left(J(k+z,\tau) + J(k-z,\tau)\right), \tag{31}$$

and ends the proof of the identity (27) and part (i) of the proposition.

Let us prove part (ii). We write

$$J(K+z,\tau) = I_1 + I_2 = \int_0^1 e^{-2x(K+z)} f_\tau(x)\mathrm{d}x + \int_1^\infty e^{-2x(K+z)} f_\tau(x)\mathrm{d}x. \tag{32}$$

Using the fact that $|z| \le 1/2$ and $|f_\tau(x)| < 1$ for $x > 0$, the second integral in the right-hand side of (32) can be bounded from above as

$$|I_2| \;<\; \int_1^\infty e^{-x(2K-1)}\mathrm{d}x = \frac{e^{-2K+1}}{2K-1} < \frac{\epsilon}{4}. \tag{33}$$

In order to deal with the first integral in the right-hand side of (32), we expand $f_\tau(x)$ in Taylor series in x (which converges for $|x| < \pi/2$, see (21)) and obtain

$$I_1 = \sum_{l\ge 0} q_l(\tau) \int_0^1 e^{-2x(K+z)} x^{2l}\mathrm{d}x = \sum_{l\ge 0} q_l(\tau) p_{2l}(2(K+z)). \tag{34}$$

In order to estimate the tail of the above series, first we will need to establish an upper bound for $q_k(\tau)$. From the following identity for Euler numbers (see formulas 9.652.3 and 9.655.3 in [1])

$$\tilde{E}_{2n} = \frac{E_{2n}}{(2n)!} = 2(-1)^n \left(\frac{2}{\pi}\right)^{2n+1} \sum_{k\ge 0} \frac{(-1)^k}{(2k+1)^{2n+1}} \tag{35}$$

we conclude that $|\tilde{E}_{2n}| < 2\left(\frac{2}{\pi}\right)^{2n+1}$. Using formula (22) and the fact that $|\tau| < 1$ we obtain

$$|q_k(\tau)| \le \sum_{j=0}^{k} |\tilde{E}_{2k-2j}| \frac{1}{j!}\left(\frac{|\tau|}{\pi}\right)^j < \sum_{j=0}^{k} 2\left(\frac{2}{\pi}\right)^{2k-2j+1} \frac{1}{j!}\left(\frac{1}{\pi}\right)^j < 2\left(\frac{2}{\pi}\right)^{2k+1} e^{\frac{\pi}{4}} < 3\left(\frac{2}{\pi}\right)^{2k}. \tag{36}$$

Using the above result we estimate the tail of the series in (34) as follows

$$\left|\sum_{l>K} q_l(\tau) p_{2l}(2(K+z))\right| < 3\sum_{l>K} \left(\tfrac{2}{\pi}\right)^{2l} < \frac{\epsilon}{4}. \tag{37}$$

Formulas (32), (33), (34) and (37) show that for all $|z| \le 1/2$ and $\tau \in (0,1)$

$$J(K+z,\tau) = \sum_{0\le l\le K} q_l(\tau) p_{2l}(2(K+z)) + \mathcal{E}_1, \tag{38}$$

where $|\mathcal{E}_1| < \frac{\epsilon}{2}$. The above statement combined with (27) gives us the desired result (28). □

Proposition 2. *There exists an algorithm such that for any $\epsilon \in (0, \frac{1}{10})$ and $x \in \mathbb{R}$ the value of $\Phi(e^{\frac{\pi i}{4}}x)$ can be evaluated to within $\pm\epsilon$ using $\le A_1 \ln\left(\frac{1}{\epsilon}\right)$ arithmetic operations on numbers of $\le A_2 \ln\left(\frac{1}{\epsilon}\right)$ bits. The algorithm requires $\le A_3 \ln\left(\frac{1}{\epsilon}\right)$ bits of memory.*

Proof. Since $\Phi(e^{\frac{\pi i}{4}}x)$ is an odd function, we can restrict x to be a positive number. When $x \in (0,1]$ we compute $\Phi(e^{\frac{\pi i}{4}}x)$ using Taylor series (25). Since this series is converging exponentially fast, it can be truncated after $O(\ln(\frac{1}{\epsilon}))$ terms in order to achieve accuracy $\pm\epsilon$.

Let us consider the case when $x \in (1,\infty)$. We will use the following result (see equations (3)-(6) in [4]): For $z \in \mathbb{C}$ with $\mathrm{Re}(z) > 0$

$$\Phi(z) = 1 - \frac{hze^{-z^2}}{\pi} \sum_{k=-\infty}^{\infty} \frac{e^{-k^2h^2}}{z^2+k^2h^2} + R(z,h) + E(z,h), \tag{39}$$

where $h > 0$ and $h\mathrm{Re}(z) \neq \pi$. Here $R(z,h) := 2(e^{\frac{2\pi z}{h}} - 1)^{-1}$ if $\mathrm{Re}(z) < \frac{\pi}{h}$ and $R(z,h) := 0$ otherwise. The error term $E(z,h)$ can be bounded from above as

$$|E(z,h)| \leq \frac{2|ze^{-z^2}|e^{-\frac{\pi^2}{h^2}}}{\sqrt{\pi}(1-e^{-2\frac{\pi^2}{h^2}})\left|(\mathrm{Re}(z))^2 - \frac{\pi^2}{h^2}\right|}. \tag{40}$$

Let us define $\gamma := \sqrt{\ln\left(\frac{1}{\epsilon}\right)}$, $z := e^{\frac{\pi \mathrm{i}}{4}}x$ and

$$h := \begin{cases} \frac{\pi}{2\gamma}, & \text{if } x \leq 2\gamma \text{ or } x \geq 4\gamma, \\ \frac{\pi}{4\gamma}, & \text{if } x \in (2\gamma, 4\gamma). \end{cases} \tag{41}$$

This choice of h implies $e^{-\frac{\pi^2}{h^2}} \leq \epsilon^4$ and

$$\frac{|z|}{\left|(\mathrm{Re}(z))^2 - \frac{\pi^2}{h^2}\right|} = \frac{x^{-1}}{\left|\frac{1}{2} - \frac{\pi^2}{x^2h^2}\right|} < 4, \tag{42}$$

therefore

$$|E(z,h)| < \frac{8\epsilon^4}{(1-\epsilon^8)} < \frac{\epsilon}{2}. \tag{43}$$

Next, we define $N = \lceil 4\gamma^2 \rceil$ and we estimate the tail of the series in (40) as

$$\left| hze^{-z^2} \sum_{n \geq N+1} \frac{e^{-k^2h^2}}{z^2+k^2h^2} \right| < h \sum_{n \geq N+1} khe^{-k^2h^2} < \int_{Nh}^{\infty} ue^{-u^2}\mathrm{d}u = \frac{1}{2}e^{-(Nh)^2} < \frac{\epsilon^4}{2} < \frac{\epsilon}{4}. \tag{44}$$

The above results show that for every $x > 1$ we can choose h according to (41) and obtain

$$\Phi\left(e^{\frac{\pi \mathrm{i}}{4}}x\right) = 1 - \frac{he^{\frac{\pi \mathrm{i}}{4}}xe^{-\mathrm{i}x^2}}{\pi}\left(-\frac{\mathrm{i}}{x^2} + 2\sum_{k=1}^{N} \frac{e^{-k^2h^2}}{\mathrm{i}x^2+k^2h^2}\right) + \mathcal{E}, \tag{45}$$

where $|\mathcal{E}| < \epsilon$. Since the number of terms in the above sum is $N = \lceil 4\ln\left(\frac{1}{\epsilon}\right)\rceil$, this ends the proof of Proposition (2) in the case $x \in (1,\infty)$. □

Proposition 3. *There exists an algorithm such that for any $\epsilon \in (0, \frac{1}{10})$, $|z| < 10$ and $\tau \in (0,1)$ the value of $h(z,\tau)$ can be evaluated to within $\pm\epsilon/\sqrt{\tau}$ using $\leq A_4 \ln\left(\frac{1}{\epsilon}\right)^2$ arithmetic operations on numbers of $\leq A_5 \ln\left(\frac{1}{\epsilon}\right)$ bits. The algorithm requires $\leq A_6 \ln\left(\frac{1}{\epsilon}\right)^2$ bits of memory.*

Proof. All computations will be performed on numbers of $A_3 \ln\left(\frac{1}{\epsilon}\right)$ bits, where A_3 is the constant from Proposition 2. We set $K = 2 + \lceil 2\ln(\frac{1}{\epsilon})\rceil$. The first step is to pre-compute and store in the memory the values of $\tilde{E}_k$ for $0 \le k \le 2K$. These numbers can be computed recursively (via formula 9.631 in [1]), this computation will require $O(K^2)$ arithmetic operations and $O(K^2)$ bits of memory. The second step is to use identity (8) and to normalize z so that $|z| \le 1/2$ (note that this will require $O(|z|)$ arithmetic operations – we will need this fact later). The thid step is to apply formula (28). According to (24) and Proposition 2, the computation of $H_K(\pm z, \tau)$ to the accuracy of $\pm\epsilon/\sqrt{\tau}$ can be achieved in $O(K^2)$ arithmetic operations using $O(K)$ bits of memory. We claim that the computation of the finite sum in (28) to the accuracy $\pm\epsilon$ can also be done in $O(K^2)$ arithmetic operations using $O(K)$ bits of memory (provided that we are using the pre-computed values of $\tilde{E}_k$). This follows from the fact that the coefficients $q_k(\tau)$ can be computed via (22) in $O(k)$ computations, while the values of $p_k(x)$ can be evaluated recursively via the identity

$$p_k(x) = \frac{1}{x}\left(kp_{k-1}(x) - e^{-x}\right), \quad k \ge 1, \tag{46}$$

which follows easily from (23) by integration by parts. Note that the above recurrence identity is numerically stable as long as $|x| \ge k$, which is true in formula (28). □

Proposition 4. *There exists an algorithm such that for any integer $n \ge 1$, $\epsilon \in (0, \frac{1}{10})$, $|z| \le 1/2$ and $|\tau| < n^{-4}$ the value of the function $F_n(z,\tau)$ can be evaluated to within $\pm\epsilon$ using $\le A_7 \ln\left(\frac{n}{\epsilon}\right)^3$ arithmetic operations on numbers of $\le A_8 \ln\left(\frac{n}{\epsilon}\right)$ bits. The algorithm requires $\le A_9 \ln\left(\frac{n}{\epsilon}\right)^2$ bits of memory.*

Proof. The main idea behind this algorithm is to expand the exponential function in Taylor series, however the details of the implementation will be different depending on whether $|z| > n^{-1}$ or $|z| < n^{-1}$. Let us consider the first case, when $|z| > n^{-1}$. We expand $\exp(2\pi \mathrm{i}\tau k^2)$ in Taylor series and obtain

$$F_n(z,\tau) = \sum_{l=0}^{\infty} \frac{(2\pi\mathrm{i})^l}{l!}\left[\sum_{k=0}^{n} (\tau k^2)^l e^{2\pi \mathrm{i} z k}\right]. \tag{47}$$

Since $|\tau| < n^{-4}$, the absolute value of the sum in the square brackets is bounded from above by $(n+1)$. Therefore, the external sum in l is converging exponentially fast, and in order to achieve accuracy $\pm\epsilon$ we can truncate it after $L \le A_{10}\ln(\frac{n}{\epsilon})$ terms for some constant A_{10}. We assume that $L^3 < n$, otherwise we will compute $F_n(z,\tau)$ by direct summation. Our goal is to show that the sum in the square brackets can be evaluated with accuracy $\pm\epsilon$ in $O(L^2)$ operations on numbers of $O(L)$ bits using $O(L^2)$ bits of memory. The main idea is to rewrite this sum as follows

$$\sum_{k=0}^{n} (\tau k^2)^l e^{2\pi \mathrm{i} z k} = \frac{\tau^l}{(2\pi\mathrm{i})^{2l}} \times \frac{\mathrm{d}^{2l}}{\mathrm{d}z^{2l}} \sum_{k=0}^{n} e^{2\pi \mathrm{i} z k} = n \frac{(\tau n^4)^l}{(2\pi\mathrm{i})^{2l}} \times n^{-4l-1} \frac{\mathrm{d}^{2l}}{\mathrm{d}z^{2l}}\left[\frac{e^{2\pi \mathrm{i} z(n+1)} - 1}{e^{2\pi \mathrm{i} z} - 1}\right]. \tag{48}$$

Using Leibniz rule we find that

$$n^{-4l-1} \frac{\mathrm{d}^{2l}}{\mathrm{d}z^{2l}}\left[\frac{e^{2\pi \mathrm{i} z(n+1)} - 1}{e^{2\pi \mathrm{i} z} - 1}\right] = \sum_{j=0}^{2l} \binom{2l}{j} f_j(z) g_{2l-j}(z), \tag{49}$$

where we have defined

$$f_j(z) := n^{-2j-1} \frac{\mathrm{d}^j}{\mathrm{d}z^j}\left[\frac{1}{e^{2\pi \mathrm{i} z} - 1}\right], \tag{50}$$

and

$$g_j(z) := n^{-2j}\frac{\mathrm{d}^j}{\mathrm{d}z^j}\left[e^{2\pi\mathrm{i}z(n+1)}-1\right] = n^{-2j}\left((2\pi\mathrm{i}(n+1))^j e^{2\pi\mathrm{i}z(n+1)} - \mathbf{1}_{\{j=0\}}\right). \tag{51}$$

While the computation of $g_j(z)$ is straighforward, the computation of $f_j(z)$ requires more work. First, we check by induction that

$$f_j(z) = \sum_{k=1}^{j+1} a_{j,k}(n(\exp(2\pi\mathrm{i}z)-1))^{-k}, \tag{52}$$

where $a_{0,1}=1$ and the remaining coefficients $a_{j,k}$ can be computed by the recursion

$$a_{j+1,k} = -\frac{2\pi\mathrm{i}}{n}\left((k-1)a_{j,k-1} + \frac{k}{n}a_{j,k}\mathbf{1}_{\{k\le j+1\}}\right), \quad j\ge 1,\ 1\le k\le j+2. \tag{53}$$

From (53) one can see by induction that $|a_{j,k}| < (4\pi j/n)^j < 1$ (recall that $j<2L<2n^{\frac{1}{3}}$). Note that the condition $n^{-1}<|z|\le 1/2$ implies

$$n|\exp(2\pi\mathrm{i}z)-1| > \max(n|\sin(2\pi z)|, n|\cos(2\pi z)-1|) \ge 4, \tag{54}$$

since $|\sin(2\pi z)|\ge 4|z|$ if $|z|\le 1/4$ and $|\cos(2\pi z)-1|>1$ if $1/4<|z|<1/2$. Given (54) and the above upper bound on the coefficients $a_{j,k}$, it is clear that formula (52) provides a numerically stable way of computing $f_j(z)$ using numbers of $O(L)$ bits. The memory requirement is $O(L^2)$ bits, since in order to compute the coefficients $a_{j+1,k}$, $1\le k\le j+2$ via (53) we need to store at most $2L+1$ numbers $a_{j,k}$, $1\le k\le j+1$.

When $|z|<n^{-1}$ the lower bound (54) is no longer valid, and we have to proceed by a different route. In this case we expand the exponential function $\exp(2\pi\mathrm{i}(zk+\tau k^2))$ in Taylor series and obtain

$$F_n(z) = \sum_{l\ge 0}\frac{(2\pi\mathrm{i})^l}{l!}\left[\sum_{k=0}^{n}(zk+\tau k^2)^l\right]. \tag{55}$$

Since $|z|<n^{-1}$ and $|\tau|\le n^{-4}$, the sum in the square brackets is bounded from above by $(n+1)2^l$. Therefore, the sum in l is converging exponentially fast, and in order to achieve accuracy $\pm\epsilon$ we can truncate it after $L = O\left(\ln\left(\frac{n}{\epsilon}\right)\right)$ terms. The sum in the square brackets in (55) can be computed as follows

$$\sum_{k=0}^{n}(zk+\tau k^2)^l = \sum_{j=0}^{l}\binom{l}{j}(nz)^{l-j}(n^2\tau)^j S_{l+j}(n), \tag{56}$$

where we have defined $S_j(n) := n^{-j}\sum_{k=0}^{n}k^j$. Formula (9.623.1) in [1] gives us

$$S_j(n) = \frac{n}{j+1}\sum_{i=0}^{j}(-1)^i\binom{j+1}{i}B_i n^{-i}, \quad j\ge 1,\ n\ge 1, \tag{57}$$

where B_i are the Bernoulli numbers $\{1,-1/2,1/6,\dots\}$. We will leave it to the reader to verify that the above three formulas provide the required algorithm for computing $F_n(z,\tau)$ to within $\pm\epsilon$ in $O(L^3)$ arithmetic operations on numbers of $O(L)$ bits (one should precompute and store $2L$ values of $B_i/i!$, $0\le i\le 2L$, which can be done in $O(L^2)$ arithmetic operations using $O(L^2)$ bits of memory). □

Proof of Theorem 2: We are given $z \in (0,1)$, $\tau \in (0,1)$, a positive integer n and a small positive number ϵ. We will describe the algorithm for computing the value of $F = F_n(z,\tau)$. In order to start the algorithm, we use identities (2) and normalize z and τ so that $|z| \leq 1/2$ and $|\tau| \leq 1/4$; we define z_1 and τ_1 to be equal to these normalized values of z and τ. The algorithm is based on a recursion, and j will be the counter which keeps track of the steps of the recursion. We initialize $j = 1$, $n_1 = n$, $\alpha_1 = 1$ and $\beta_1 = 0$. All computations are performed on numbers of $\lceil \max(5A_5, 3A_8) \ln\left(\frac{n}{\epsilon}\right) \rceil$ bits, where A_5 and A_8 are the constants appearing in Propositions 3 and 4.

The algorithm:

(i) If $n_j \leq \ln(n)^3$ then we compute $f = F_{n_j}(z_j, \tau_j)$ by direct summation. Terminate the algorithm and return $F = \alpha_j f + \beta_j$.

(ii) If $|\tau_j| < n_j^{-4}$ then we compute $f = F_{n_j}(z_j, \tau_j)$ to the accuracy of $\pm\epsilon/n^3$ using the algorithm from Proposition 4. Terminate the algorithm and return $F = \alpha_j f + \beta_j$.

(iii) If $|\tau_j| \geq n_j^{-4}$ we set $n_{j+1} = \lfloor 2n_j|\tau_j| \rfloor$, $\tilde{z} = \frac{z_j}{2|\tau_j|}$ and $\tilde{\tau} = -\frac{1}{4\tau_j}$. Set z_{j+1} and τ_{j+1} to be the normalized values of $\tilde{z}$ and $\tilde{\tau}$ (use identities (2)). If $\tau_j > 0$, then

$$\alpha_{j+1} = \frac{\alpha_j}{\sqrt{2\tau_j}} \exp\left(\frac{\pi \mathrm{i}}{4} - \frac{\pi \mathrm{i} z_j^2}{\tau_j}\right), \quad \beta_{j+1} = \beta_j + \alpha_j R_{n_{j+1},n_j}(z_j, \tau_j), \tag{58}$$

while if $\tau_j < 0$ then

$$\alpha_{j+1} = \frac{\alpha_j}{\sqrt{2|\tau_j|}} \exp\left(-\frac{\pi \mathrm{i}}{4} - \frac{\pi \mathrm{i} z_j^2}{\tau_j}\right), \quad \beta_{j+1} = \beta_j + \alpha_j \overline{R_{n_{j+1},n_j}(-z_j, |\tau_j|)}, \tag{59}$$

where $R_{m,n}(z,\tau)$ is given by by (7), and the values of the Mordell integral $h(\cdot,\cdot)$ appearing in (7) are computed using the algorithm from Proposition 3 to the accuracy $\pm\epsilon/n^3$.

(iv) Increase the counter $j \mapsto j+1$ and proceed to step (i).

Assume that this algorithm stops after J iterations. The fact that this algorithm returns the correct value of $F_n(z,\tau)$ can be verified by induction on J using identity (6). Let us investigate the number of arithmetic operations required by this algorithm. At each iteration of the algorithm, provided that it does not terminate in steps (i) or (ii), we have the new value n_{j+1} which satisfies $n_{j+1} = \lfloor 2n_j|\tau_j| \rfloor \leq n_j/2$ (recall that $|\tau_j| \leq 1/4$). This shows that the algorithm will either terminate in step (i) after at most $\log_2(n)$ iterations, or it will terminate in step (ii) before that. Let us denote $L = \ln\left(\frac{n}{\epsilon}\right)$. Step (ii) (resp. step (iii)) requires $O(L^3)$ (resp. $O(L^2)$) arithmetic operations and both of these steps require $O(L^2)$ bits of memory (see Propositions 3 and 4). Since step (ii) will be executed at most once, and step (iii) at most $\log_2(n)$ times, it is clear that the algorithm requires $O(L^3)$ arithmetic operations and $O(L^2)$ bits of memory.

Finally, let us consider the accuracy of this algorithm. All numbers appearing in the algorithm are evaluated to the accuracy $\pm\epsilon/n^3$. Since the algorithm did not terminate at the iteration $J-1$, we have $n_{J-1} > 1$ and $|\tau_{J-1}| \geq n_{J-1}^{-4} > n^{-4}$. Recall that for all j we have $n_{j+1} \leq 2n_j|\tau_j|$, therefore

$$1 < n_{J-1} \leq 2^{J-2} n \prod_{i=1}^{J-2} |\tau_i|, \tag{60}$$

and applying formulas (58) and (59) we obtain

$$|\alpha_{J-1}| = \left[2^{J-2} \prod_{i=1}^{J-2} |\tau_i| \right]^{-\frac{1}{2}} < \sqrt{n}. \tag{61}$$

Assuming that the algorithm terminates in step (ii), then the final accuracy is at least $(\pm\epsilon/n^3) \times \sqrt{n} \times \log_2(n)$, which is smaller than the required accuracy $\pm\epsilon$. On the other hand, if the algorithm terminates in step (i), then $|\alpha_J| = |\alpha_{J-1}|/\sqrt{2|\tau_{J-1}|} < n^{\frac{5}{2}}$, and the final accuracy is $(\pm\epsilon/n^3) \times n^{\frac{5}{2}} \times \log_2(n)$, which is still smaller than $\pm\epsilon$. □

Remark 1: One may ask the following natural question: is the choice $n_{j+1} = \lfloor 2n_j|\tau_j| \rfloor$ in the above algorithm optimal? In other words, given that the identity (6) is true for all positive m and n, why can not we choose $m = n_{j+1} \ll 2n_j|\tau_j|$, thus reducing the number of terms in the new sum $F_m(\cdot,\cdot)$? The rationale for this choice is that the computation $R_{m,n_j}(\cdot,\cdot)$ in formula (7) requires the evaluation of the Mordell integral

$$h(z_j + (2n_j+1)|\tau_j| - m - \tfrac{1}{2}, -2\tau_j),$$

which involves more than $|2n_j|\tau_j| - m|$ arithmetic operations (see step 2 in the proof of Proposition 3). Therefore, while the choice of $m = n_{j+1} \ll 2n_j|\tau_j|$ will decrease the number of terms (and the computation time) of $F_m(\cdot,\cdot)$, any gain will be canceled by the corresponding increase in the computation time of $R_{m,n_j}(\cdot,\cdot)$.

3 Practical implementation and extensions of the algorithm

As is often the case, the algorithm which can be analyzed analytically and which allows for rigorous error bounds is not necessarily the most efficient algorithm from the practical point of view. While it is certainly possible to perform practical computations of $F_n(z,\tau)$ using the algorithm described in the proof of Theorem 2, in this section we will explain how one could produce a more efficient algorithm with a certain amount of pre-computation and a few numerical experiments. This practical implementation is suitable in the case when we need to compute $F_n(z,\tau)$ to a fixed accuracy $\pm\epsilon$ for many different values of z, τ and $n \le N_1$ (for some fixed value of N_1).

As we saw in the proof of Theorem 2, the main computational effort is spent in evaluating $F_n(z,\tau)$ for very small values of τ (when $|\tau| < n^{-4}$) and in computing the Mordell integral $h(z,\tau)$. We do not see a way of making the former of these tasks much faster, however the latter task can certainly be done much more efficiently. Part (i) of Proposition 1 shows that in order to compute $h(z,\tau)$ we need to be able to evaluate the error function $\Phi(e^{\frac{\pi \mathrm{i}}{4}}x)$ for $x \in \mathbb{R}$ and to compute $J(z,\tau)$ defined by (26). Let us first discuss the computation of the error function. Our approach to computing $\Phi(e^{\frac{\pi \mathrm{i}}{4}}x)$ is to divide the interval $(0,\infty)$ into a number of sub-intervals $0 < x_1 < x_2 < \cdots < x_m = x^* < \infty$, and use the Chebyshev approximation on each sub-interval. On the infinite interval (x^*,∞) we define the function $f(\sigma)$ via $\Phi(e^{\frac{\pi \mathrm{i}}{4}}x) = 1 + e^{-\mathrm{i}xz^2}x^{-1}f(\sigma)$, where $\sigma := (x^*/x)^2$ and we approximate $f(\sigma)$ by the first few terms of the Chebyshev series. It is known (see [8]) that the coefficients of the corresponding Chebyshev series decay as $\exp(-2\sqrt{nx^*})O(n^{-\frac{1}{2}})$, therefore, by a proper choice of x^* we can be sure that we need only a few terms of the Chebyshev series to obtain the required accuracy. Once we have chosen x^*, we divide $(0,x^*)$ into m sub-intervals of equal length, and on each of them we compute the first few terms of the Chebyshev series. Note that on each finite interval (x_i,x_{i+1}), $1 \le i < m$, the Chebyshev series approximating $\Phi(e^{\frac{\pi \mathrm{i}}{4}}x)$ must converge exponentially fast since $\Phi(z)$ is an entire functions. By choosing

m large enough we can make sure that the number of significant terms in each Chebyshev series is small. For example, in our implementation of this algorithm we chose $x^* = 5$ and $m = 5$, and on each subinterval we approximated $\Phi(e^{\frac{\pi \mathrm{i}}{4}} x)$ by the first thirty terms of Chebyshev series. This approximation had absolute error $\leq 10^{-30}$ over all real values of x.

The second important problem is how to evaluate $J(k+z,\tau)$ efficiently. Our approach is based on the following formula

$$J(k+z,\tau) = \frac{1}{2k} \int_0^\infty e^{-y} g_{z,\tau}\left(\tfrac{y}{k}\right) \mathrm{d}y, \quad \text{where} \quad g_{z,\tau}(y) := \frac{e^{\frac{\mathrm{i}\tau y^2}{4\pi} - zy}}{\cosh(\frac{y}{2})}, \tag{62}$$

which follows from (26) by changing the variable of integration $x = y/(2k)$. For $k \geq 1$, $|z| \leq 1/2$ and $\tau \in (0,1)$ the function $y \in (0,\infty) \mapsto g_{z,\tau}\left(\frac{y}{k}\right)$ is bounded, and as $k \to +\infty$ it converges to $g_{z,\tau}(0) = 1$. Therefore, when k is reasonably large, the integral in (62) can be computed to a very high-precision using the Gauss-Laguerre quadrature with the weight function e^{-y} and M nodes. The problem is to decide what "reasonably large" means, and here one should do a number of numerical experiments to find the optimal values of k and M. If we take k to be a large number, then the function $g_{z,\tau}\left(\frac{y}{k}\right)$ is very close to 1, and M – the number of nodes in Gauss-Laguerre quadrature – can be taken quite small in order to achieve the required accuracy. Of course, the disdvantage of choosing k to be large is that it would require many evaluations of the error function in (24), and it would increase the run-time of the algorithm. On the other hand, if we take k to be a small integer, then the function $g_{z,\tau}\left(\frac{y}{k}\right)$ oscillates and M has to be very large in order to provide the required accuracy, which would again increase the run-time of the algorithm. Therefore, k and M have to be chosen so that the computation time of $H_k(z,\tau)$ it approximately equal to the computation time of $J(k+z,\tau)$.

In our examples we took $k = 5$ in formulas (27), (24) and (26) and we have used the Gauss-Laguerre quadrature with $M = 124$ nodes (truncated to the smallest 40 nodes, see [5]) to evaluate the integral in (26). In order to verify the accuracy, we have computed $h(z,\tau)$ on a very fine regular grid of points in the rectangle $|z| \leq 1/2$ and $\tau \in (0,1/2)$ using the above algorithm and we have checked that the relative error is always less than 10^{-29} (when compared with the algorithm described in Proposition 3). While the algorithm based on Gauss-Laguerre quadrature is very efficient, we were not able to provide rigorous error estimates. It is known that the error of the M-point Gauss-Laguerre quadrature can be bounded by a multiple of

$$\eta_{2M} := \sup\left\{ \left| \frac{\partial^{2M}}{\partial y^{2M}} g_{z,\tau}(y) \right| \ : \ y \geq 0 \right\}, \tag{63}$$

(see Theorem 3 in [10]), but we were not able to obtain good upper bounds for this quantity.

The results of our numerical experiments are presented in Figure 1. The algorithm was implemented in Fortran using the quadruple precision. Due to the fact that the number of iterations of the algorithm varies for different values of τ, we have tested the algorithm for 1000 random pairs (z,τ), sampled uniformly from the domain $0 < \tau < 1/4$ and $-1/2 < z < 1/2$, and the results in Figure 1 represent the average run-time for a single evaluation of $F_n(z,\tau)$. Since we are working in fixed precision (which does not depend on n, as in the algorithm in the proof of Theorem 2), these computations involve an unavoidable loss of precision, which becomes more pronounced as n increases. See the paragraph preceding Remark 1 on page 12, explaining the reason for this loss of precision. Our results indicate that the difference between the values of $F_n(z,\tau)$ computed via direct summation (1) and our version of Hiary's algorithm is typically of the order of 10^{-28} for $n = 10^3$ and around 10^{-25} for $n = 10^5$. This loss of precision is still acceptable for practical purposes. Most importantly, the figures 1b and 1d confirm

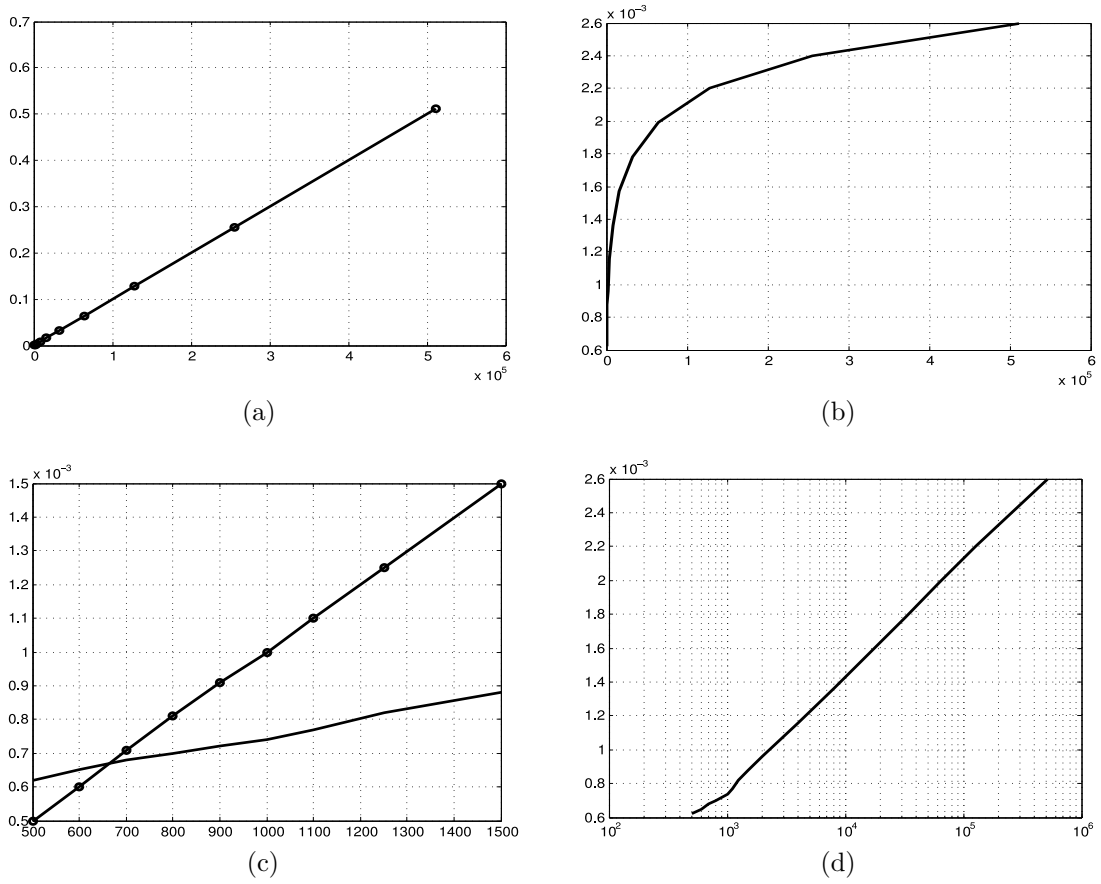

Figure 1: Average run time for computing a single value of $F_n(z,\tau)$ (x-axis represents n, y-axis represents time in seconds). The curve marked with circles corresponds to direct summation algorithm based on (1), the curve without markers corresponds to Hiary's algorithm.

that the run-time of the algorithm increases essentially logarithmically with n, and that the simplified version of Hiary's algorithm is much faster than the summation of n terms in (1) even for moderately large n.

Finally, we will discuss a related problem of evaluating the finite sum of the form

$$F_{n,j}(z,\tau) := n^{-j} \sum_{k=1}^{n} k^j e^{2\pi \mathrm{i}(zk+\tau k^2)} = (2\pi \mathrm{i} n)^{-j} \frac{\partial^j}{\partial z^j} F_n(z,\tau). \tag{64}$$

Hiary's result [3, Theorem 1.1] states that $F_{n,j}(z,\tau)$ can be computed to the required accuracy $\pm\epsilon$ in poly-log time in n/ϵ (though the precise statement is slightly more complicated, as the implied constants also depend on j). Our results can also be adapted to give a simpler version of Hiary's poly-log time algorithm for computing $F_{n,j}(z,\tau)$. Below we will sketch the main ideas of the practical implementation of such an algorithm.

We will follow the same path as in the algorithm described in the proof of Theorem 2. Our goal is to compute the values of $F_{n,j}(z,\tau)$ for $0 \le j \le J$. At each step of the recursion, we normalize the values of z and τ so that $|z| \le 1/2$ and $|\tau| \le 1/4$. If $|\tau| < n^{-4}$, then we compute $F_{n,j}(z,\tau)$ by expanding the exponential function in (64) in Taylor series and applying similar ideas as in the proof of Proposition 4.

If $|\tau| \geq n^{-4}$, then we use the following identities

$$\frac{\partial^j}{\partial z^j} F_n(z,\tau) = \frac{e^{\frac{\pi \mathrm{i}}{4}}}{\sqrt{2\tau}} \sum_{k=0}^{j} \binom{j}{k} \left[\frac{\partial^{j-k}}{\partial z^{j-k}} e^{-\frac{\pi \mathrm{i} z^2}{2\tau}}\right] \times \left[\frac{\partial^k}{\partial z^k} F_m\left(\tfrac{z}{2\tau}, -\tfrac{1}{4\tau}\right)\right] + \frac{\partial^j}{\partial z^j} R_{m,n}(z,\tau), \quad 0 \leq j \leq J, \quad (65)$$

which follow from (6) by applying Leibniz rule. These identities reduce the computation of $F_{n,j}(z,\tau)$ to the computation of $F_{m,j}(\cdot,\cdot)$ with $m = \lfloor 2n\tau \rfloor < n/2$ and $1 \leq j \leq J$, and complete the main step of the recursion.

Applying identities (65) in practice will involve computing $\frac{\partial^j}{\partial z^j} h(z,\tau)$, which is equivalent to evaluating $G_1 = \frac{\partial^j}{\partial z^j} H_k(z,\tau)$ and $G_2 = \frac{\partial^j}{\partial z^j} J(k+z,\tau)$ (see formulas (24), (26) and (27)). Computing G_1 does not pose a serious problem, as applying Leibniz rule to (24) would give us an explicit expression for G_1, and since $\Phi'(z) = (2/\sqrt{\pi})\exp(-z^2)$ it is easy to see that this explicit expression would involve only elementary functions and the error function $\Phi(z)$. At the same time, when z is large it will be more efficient to compute $\frac{\partial^j}{\partial z^j} e^{z^2}(\Phi(z)-1)$ directly by taking derivatives of the right-hand side of formula (45) (or by using the asymptotic expansion for this function, see formula 8.254 in [1]). The value of $G_2 = \frac{\partial^j}{\partial z^j} J(k+z,\tau)$ can be computed using the generalized Gauss-Laguerre quadrature. Indeed, from (26) we find that

$$\frac{\partial^j}{\partial z^j} J(k+z,\tau) = \frac{(-1)^j}{2} k^{-j-1} \int_0^\infty e^{-y} y^j g_{z,\tau}\left(\tfrac{y}{k}\right) \mathrm{d}y, \quad (66)$$

where $g_{z,\tau}(y)$ is defined in (62). Using the same strategy as discussed on page 13 (following equation (62)) the integral in the right-hand side of (66) can be evaluated using the generalized Gauss-Laguerre quadrature with the weight function $x^j e^{-x}$ and M nodes. By experimenting with different values of k and M (and choosing the optimal ones) we can compute $\frac{\partial^j}{\partial z^j} h(z,\tau)$ very efficiently with the required accuracy.

References

[1] I. S. Gradshteyn and I. M. Ryzhik. *Table of integrals, series, and products*. Elsevier/Academic Press, Amsterdam, seventh edition, 2007.

[2] G. A. Hiary. Fast methods to compute the Riemann zeta function. *Ann. of Math.*, 174(2):891–946, 2011.

[3] G. A. Hiary. A nearly-optimal method to compute the truncated theta function, its derivatives, and integrals. *Ann. of Math.*, 174(2):859–889, 2011.

[4] D. Hunter and T. Regan. A note on the evaluation of the complementary error function. *Math. Comp.*, 26(118):539–541, 1972.

[5] G. Mastroianni and G. Monegato. Some new applications of truncated Gauss-Laguerre quadrature formulas. *Numerical Algorithms*, 49(1-4):283–297, 2008.

[6] L. J. Mordell. The value of the definite integral $\int_{-\infty}^{\infty} \frac{e^{at^2+bt}}{e^{ct}+d} dt$. *Quarterly Journal of Math.*, 68:329–342, 1920.

[7] L. J. Mordell. The definite integral $\int_{-\infty}^{\infty} \frac{e^{at^2+bt}}{e^{ct}+d} dt$ and the analytic theory of numbers. *Acta Math.*, 61:322–360, 1933.

[8] G. Nemeth. Chebyshev expansions for Fresnel integrals. *Numerische Mathematik*, 7:320–312, 1965.

[9] S. Ramanujan. Some definite integrals connected with Gauss sums. *Messenger of Mathematics*, 44:75–85, 1915.

[10] A. H. Stroud and K. W. Chen. Peano error estimates for Gauss-Laguerre quadrature formulas. *SIAM J. Numer. Anal.*, 9(2):333–339, 1972.

[11] E. Titchmarsh. *The theory of the Riemann zeta-function*. Oxford University Press, second edition, revised by D. R. Heath-Brown, 1986.

[12] W. Van Snyder. Algorithm 723: Fresnel integrals. *ACM Trans. Math. Softw.*, 19(4):452–456, Dec. 1993.

[13] S. Zwegers. Mock theta functions. Ph.D. thesis, Utrecht University, arXiv:0807.4834, 2002.