# Polylogarithms and the BBP Algorithm

Emma Cardwell

March, 2020

## 1 Introduction to Polylogarithms

The polylogarithm is family of functions of complex numbers. It is sometimes referred to as the Jonquière function, in honor of mathematician Alfred Jonquière. Special cases of the polylogarithm function, such as the dilogarithm and the trilogarithm, have been studied by mathematicians since the 1800s [DLMF, Section 25.12]. The polylogarithm defines a Taylor series expansion for the general form of logarithms, extended to the complex plane.

**Definition 1.** *The polylogarithm function,* $\mathrm{Li}_s(z)$ *is defined as*

$$\mathrm{Li}_s(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^s} = z + \frac{z^2}{2^s} + \frac{z^3}{3^s} + ... \quad \textit{for } z \in \mathbb{C}, |z| < 1 \tag{1}$$

This is sometimes referred to as the classical polylogarithm. The classical polylogarithm can also be expressed with integrals. The integral representations often extend the domain of the classical polylogarithm function. They define a radius of convergence for the function that is larger than $|z| < 1$. Various integral representations of polylogarithms can be used to describe the Fermi-Dirac distribution (which models the distribution of particles over energy states) and the Maxwell-Boltzmann distribution (which models the distribution of ideal gas particles' speeds).

## 2 Special Cases of the Polylogarithm Function

### Natural Logarithm

When $s = 1$, the polylogarithm function defines a variant of the natural logarithm:

$$\mathrm{Li}_1(z) = \sum_{k=1}^{\infty} \frac{z^k}{k} = z + \frac{z^2}{2} + \frac{z^3}{3} + ... \quad \text{for } z \in \mathbb{C}, |z| < 1$$

This sum is equal to the Taylor expansion for $-\ln(1-x)$. We can find $-\ln(1-x)$ by integrating the Taylor expansion for $\frac{1}{1-x}$:

$$\begin{aligned}
\int \frac{1}{1-x}dx &= \int \sum_{n=0}^{\infty} x^n dx \quad \text{for } x \in \mathbb{R}, |x| < 1 \\
-\ln(1-x) &= \int dx + \int x dx + \int x^2 dx + \int x^3 dx ... \\
-\ln(1-x) &= x + \frac{x^2}{2} + \frac{x^3}{3} + \frac{x^4}{4} ... \\
-\ln(1-x) &= \sum_{k=1}^{\infty} \frac{x^k}{k} \quad \text{for } x \in \mathbb{R}, |x| < 1
\end{aligned}$$

The only issue is, the logarithm function here is defined over $\mathbb{R}$. However, the natural logarithm function can be extended to the complex plane. For a complex number $z = a + bi$ we can convert it to polar form and then use Euler's formula to change that into an exponential function:

$$\begin{aligned}
z &= |z|(\cos\theta + i\sin\theta) \\
z &= |z|e^{i\theta}
\end{aligned}$$

Where $|z| = \sqrt{a^2+b^2}$ and $\theta = \begin{cases} 2\arctan\left(\frac{b}{\sqrt{a^2+b^2}+a}\right) & \text{if } a > 0 \text{ or } b \not\equiv 0 \\ \pi & \text{if } a < 0 \text{ and } b = 0 \\ \text{undefined} & \text{if } a = b = 0 \end{cases}$

Note that, since sine and cosine are periodic with period $2\pi$,

$$\begin{aligned}
z = |z|(\cos\theta + i\sin\theta) &= |z|\big(\cos(\theta + 2\pi k) + i\sin(\theta + 2\pi k)\big) \quad \text{for all } k \in \mathbb{Z} \\
|z|e^{i\theta} &= |z|e^{i(\theta+2\pi k)}
\end{aligned}$$

This means that, when we take the logarithm of a complex number, we get a multi-valued output:

$$\begin{aligned}
\ln z &= \ln(|z|e^{i(\theta+2\pi k)}) \quad \text{for all } k \in \mathbb{Z} \\
\ln z &= \ln|z| + i(\theta + 2\pi k)
\end{aligned}$$

We can restrict the output by choosing our value for $\theta$ such that $-\pi < \theta < \pi$. This restricts the function to the cut plane $\mathbb{C} - (-\infty, 0)$. Now we can restrict $-\ln(1-z)$:

$$-\ln(1-z) = -ln|1-z| - i\theta \quad z \in \mathbb{C} - (1, \infty)$$

$-\ln(1-z)$ will give us a single-valued output when $|z| < 1$.
This shows how we can extend the definition of the logarithm function to the complex plane.

### Multiple Polylogarithms

**Definition 2.** *The multiple polylogarithm (also referred to as the multidimensional polylogarithm) function in a single variable,* $\text{Li}_{s_1,\ldots,s_m}(z)$ *is defined as*

$$\text{Li}_{s_1,\ldots,s_k}(z) = \sum_{n_1>n_2>\ldots>n_k\geq 1} \frac{z^{n_1}}{n_1^{s_1}\ldots n_k^{s_k}} \quad \textit{for } s_1,\ldots,s_k \in \mathbb{Z}, z\in\mathbb{C}, |z|<1 \tag{2}$$

When $k=1$, the multiple polylogarithm function is equivalent to the classical polylogarithm defined earlier, $\text{Li}_s(z)$.

We can further extend this by defining the multiple polylogarithm function for multiple variables.

**Definition 3.** *The multiple polylogarithm function for multiple variables,* $\text{Li}_{s_1,\cdots,s_m}(z_1,\cdots,z_k)$ *is defined as*

$$\lambda\begin{pmatrix} z_1,\cdots,z_k \\ s_1,\cdots,s_k\end{pmatrix} = Li_{s_1,\cdots,s_m}(z_1,\cdots,z_k) = \sum_{n_1>n_2>\cdots>n_k\geq 1}\prod_{j=1}^{k}\frac{z_j^{n_j}}{n_j^{s_j}} \tag{3}$$

*for* $z_1,\cdots,z_k\in\mathbb{C},\ |z_1|,\cdots,|z_k|<1,\ s_1,\cdots s_k\in\mathbb{Z}$

For example,

$$\text{Li}_{1,3,17}(z_1,z_2,z_3) = \sum_{n_1>n_2>n_3\geq 1}\prod_{j=1}^{3}\frac{z_j^{n_j}}{n_j^{s_j}} = \sum_{n_1>n_2>n_3\geq 1}\frac{(z_1)^{n_1}(z_2)^{n_2}(z_3)^{n_3}}{(n_1)^1(n_2)^3(n_3)^{17}}$$

Here are some terms of $\text{Li}_{1,3,17}(z_1,z_2,z_3)$ written out:

$$\begin{aligned}\text{Li}_{1,3,17}(z_1,z_2,z_3) = {} & \frac{(z_1)^3(z_2)^2(z_3)^1}{(3)^1(2)^3(1)^{17}} + \frac{(z_1)^4(z_2)^2(z_3)^1}{(4)^1(2)^3(1)^{17}} + \frac{(z_1)^5(z_2)^2(z_3)^1}{(5)^1(2)^3(1)^{17}} + \cdots \\ & + \frac{(z_1)^4(z_2)^3(z_3)^1}{(4)^1(3)^3(1)^{17}} + \frac{(z_1)^5(z_2)^3(z_3)^1}{(5)^1(3)^3(1)^{17}} + \frac{(z_1)^6(z_2)^3(z_3)^1}{(6)^1(3)^3(1)^{17}} + \cdots \\ & + \cdots \\ & + \frac{(z_1)^4(z_2)^3(z_3)^2}{(4)^1(3)^3(2)^{17}} + \frac{(z_1)^5(z_2)^3(z_3)^2}{(5)^1(3)^3(2)^{17}} + \frac{(z_1)^6(z_2)^3(z_3)^2}{(6)^1(3)^3(2)^{17}} + \cdots \\ & + \cdots\end{aligned}$$

When $s=0$, the multiple polylogarithm function for multiple variables is equal to the classical polylogarithm, $\text{Li}_s(z)$.

For multiple polylogarithms with multiple variables, an interesting question to ask is how particular functions can be simplified or represented differently. We can define the *depth* of a multiple polylogarithm to be $k$, and the *weight* to be the sum of $s_1+\cdots+s_k$. To rephrase our question, we want to figure out which sums we can reduce by representing them as some combination of lower depth sums. If the original polylogarithm can be expressed entirely with depth-1 sums, then we say that it *evaluates*. There exist sums which can't be reduced, but it is very difficult to prove irreducibility [BBP97].

# 3 Connections to Other Functions

### Relation to the Riemann-Zeta Function

When $z = 1$, the polylogarithm defines the single-value Riemann-zeta function:

$$\mathrm{Li}_s(1) = \sum_{k=1}^{\infty} \frac{1}{k^s} = \zeta(s)$$

A specific case of both the multiple polylogarithm function and the multiple polylogarithm function for multiple variables is the multiple zeta function:

$$\mathrm{Li}_{s_1,\cdots,s_k}(1,\cdots,1) = \mathrm{Li}_{s_1,\cdots,s_k}(1) = \sum_{n_1>\cdots n_k\geq 1} \frac{1}{(n_1)^{s_1}\cdots(n_k)^{s_k}} = \zeta(s_1,\cdots,s_k)$$

### Relation to the Lerch Zeta Function

The polylogarithm is a special case of the Lerch zeta function.

**Definition 4.** *The Lerch zeta function, $\phi(x,a,s)$, is defined as*

$$\phi(x,a,s) = \sum_{n=0}^{\infty} \frac{e^{2n\pi ix}}{(a+n)^s}$$

*for $s, x \in \mathbb{R}$, $s > 1$, and $a$ not equal to a negative integer or zero.*

When $a = 0$, we get the Hurwitz zeta function $\zeta(s,c) := \sum_{n=0}^{\infty} \frac{1}{(n+c)^s}$, which generalizes the Riemann-zeta function. When $s = 1$, the Lerch zeta function is related to polylogarithms. Let $z = e^{2\pi ix}$. Then, the Lerch zeta function looks quite similar to the polylogarithm [LL10]:

$$\begin{aligned}\phi(x,a,1) &= \sum_{n=0}^{\infty} \frac{z^n}{(n+1)^s} \\ &= \sum_{n=1}^{\infty} \frac{z^{n-1}}{n^x}\end{aligned}$$

# 4 The BBP Algorithm

The Bailey-Borwein-Plouffe (BBP) computes digits of $\pi$ in hexadecimal without computing the previous terms. All BBP-type algorithms are spigot algorithms, which can compute digits of irrational numbers without calculating the previous digits and many BBP-type algorithms rely on polylogarithmic ladders (identities involving polylogarithms).

To demonstrate how BBP-type algorithms work, we can examine a simple BBP-type algorithm for calculating the digits of log 2 in binary. Borwein and Plouffe noticed that we can use a series expansion for log 2 ($\log 2 = \sum_{k=0}^{\infty} \frac{1}{k2^k}$) to calculate its binary digits from any starting position. Let $[n]$ denote the fractional part of $n$. If we wish to find the digits of log 2 after the first $d$ binary digits, we can calculate $[2^d \log 2]$. The first digits of this should be what we're looking for. We can rewrite $[2^d \log 2]$ with our previously defined expansion for log 2:

$$\begin{aligned}
[2^d \log 2] &= \left[2^d \cdot \sum_{k=1}^{\infty} \frac{1}{k2^k}\right] \\
&= \left[\sum_{k=1}^{\infty} \frac{2^{d-k}}{k}\right] \\
&= \left[\left[\sum_{k=1}^{d} \frac{2^{d-k}}{k}\right] + \sum_{k=d+1}^{\infty} \frac{2^{d-k}}{k}\right] \\
&= \left[\left[\sum_{k=1}^{d} \frac{2^{d-k} \mod k}{k}\right] + \sum_{k=d+1}^{\infty} \frac{2^{d-k}}{k}\right]
\end{aligned}$$

To provide a quick overview of how the algorithm is applied, the first of the two sums in the equation above has $d$ terms. This can be computed relatively quickly by a computer using the binary algorithm for exponentiation modulo $k$. The terms in the second sum quickly become very small (and therefore negligible). We only need to calculate the first couple of terms, or until the error doesn't affect the accuracy we want.

**Binary algorithm for exponentiation (exponentiation by squaring)**: First, lets review what the binary algorithm for exponentiation is: when computing $a^n$, multiplying $a$ by itself $n$ times is inefficient for large values of $n$. Instead, we can convert $n$ into its binary representation. We can separate $a^n$ such that it is represented by a product of $a$'s raised to powers of 2. Now, to calculate $a^n$, we only need to compute $a^1, a^2, a^4, a^8, ...a^{\lfloor \log_2 n \rfloor}$, and then multiply together the right powers of $a$ to find $a^n$.

**Binary algorithm for exponentiation modulo** $k$: Let $r = b^n \mod k$, where $r, b, n, k \in \mathbb{Z}^+$. First, we define $t$ as the largest power of 2 that is less than $n$ and let $r = 1$. Then, if $n \geq t$, we let $r = br \mod k$ and $n = n - t$. If, after redefining these parameters, $t \geq 1$, then re-define $t$ and repeat this process. This method of computing exponents allows the calculation to be performed in a shorter amount of time and also requires less computer memory [Bai06].

Now, the actual BBP algorithm! Here it is:

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right)$$

This formula was found by a computer running an integer relation-finding algorithm, but we can also prove it.

**Lemma 1.** *For any* $j \geq 1$,

$$\int_0^{\frac{1}{\sqrt{2}}} \frac{x^{j-1}}{1-x^8} dx = \left(\frac{1}{\sqrt{2}}\right)^j \sum_{k=0}^{\infty} \frac{1}{16^k(8k+j)}$$

*Proof.* To prove Lemma 1, we start with the series expansion of $\frac{1}{1-u}$:

$$\frac{1}{1-u} = 1 + u + ... = \sum_{k=0}^{\infty} u^k$$

$$\text{let } u = x^8 : \qquad \frac{1}{1-x^8} = \sum_{k=0}^{\infty} x^{8k}$$

$$\text{multiply by } x^{j-1} : \qquad \frac{x^{j-1}}{1-x^8} = \sum_{k=0}^{\infty} x^{j-1+8k}$$

Then, we integrate both sides of the equation above:

$$\begin{aligned}
\int_0^{\frac{1}{\sqrt{2}}} \frac{x^{j-1}}{1-x^8} dx &= \int_0^{\frac{1}{\sqrt{2}}} \left( \sum_{k=0}^{\infty} x^{j-1+8k} \right) dx \\
&= \sum_{k=0}^{\infty} \left( \frac{x^{j+8k}}{j+8k} \right) \Big]_0^{\frac{1}{\sqrt{2}}} \\
&= \sum_{n=0}^{\infty} \frac{\left(\frac{1}{\sqrt{2}}\right)^{j+8k}}{j+8k} - 0 \\
&= \left( \frac{1}{\sqrt{2}} \right)^j \sum_{n=0}^{\infty} \frac{1}{16^k (j+8k)}
\end{aligned}$$

□

We can use our result from Lemma 1 to prove the BBP formula for $\pi$:

$$\begin{aligned}
\sum_{k=0}^{\infty} \frac{1}{16^k} \left( \frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right) &= 4 \sum_{k=0}^{\infty} \frac{1}{16^k(8k+1)} - 2 \sum_{k=0}^{\infty} \frac{4}{16^k(8k+4)} \\
&\quad - \sum_{k=0}^{\infty} \frac{4}{16^k(8k+5)} - \sum_{k=0}^{\infty} \frac{4}{16^k(8k+6)} \\
&= 4\sqrt{2} \int_0^{\frac{1}{\sqrt{2}}} \frac{x^{1-1}}{1-x^8} dx - 2(\sqrt{2})^4 \int_0^{\frac{1}{\sqrt{2}}} \frac{x^{4-1}}{1-x^8} dx \\
&\quad - (\sqrt{2})^5 \int_0^{\frac{1}{\sqrt{2}}} \frac{x^{5-1}}{1-x^8} dx - (\sqrt{2})^6 \int_0^{\frac{1}{\sqrt{2}}} \frac{x^{6-1}}{1-x^8} dx \\
&= \int_0^{\frac{1}{\sqrt{2}}} \frac{4\sqrt{2} - 8x^3 - 4\sqrt{2}x^4 - 8x^5}{1-x^8} dx
\end{aligned}$$

To simplify this integral, we can make the substitution $u = \sqrt{2}x$:

$$\int_0^{\frac{1}{\sqrt{2}}} \frac{4\sqrt{2} - 8x^3 - 4\sqrt{2}x^4 - 8x^5}{1-x^8} dx = \int_0^1 \frac{16u - 16}{u^4 - 2u^3 + 4u - 4} du$$

We could evaluate this integral using partial fractions, or we can just trust Wolfram Alpha :) Either way, it evaluates to $\pi$ [BBP97]. ■

Now, the fun part! Let's actually find some digits of $\pi$! To find the $(d+1)^{\text{th}}$ digit of $\pi$, we need to find $[16^d\pi]$. We use 16 instead of 2 here because we want the hexadecimal digit, not the binary digit. Using the BBP formula, we have:

$$\begin{aligned}
[16^d\pi] &= \left[16^d \cdot \sum_{k=0}^{\infty} \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right)\right] \\
&= \left[4 \cdot \sum_{k=0}^{\infty} \frac{16^{d-k}}{8k+1} - 2 \cdot \sum_{k=0}^{\infty} \frac{16^{d-k}}{8k+4} - \sum_{k=0}^{\infty} \frac{16^{d-k}}{8k+5} - \sum_{k=0}^{\infty} \frac{16^{d-k}}{8k+6}\right]
\end{aligned}$$

Now, we can compute the fractional bits of the four summations separately. For example, looking at the first summation from the equation above:

$$\begin{aligned}
4 \cdot \sum_{k=0}^{\infty} \frac{16^{d-k}}{8k+1} &= \left[\left[\sum_{k=0}^{d} \frac{16^{d-k}}{8k+1}\right] + \sum_{k=d+1}^{\infty} \frac{16^{n-k}}{8k+1}\right] \\
&= \left[\left[\sum_{k=0}^{d} \frac{16^{d-k} \bmod (8k+1)}{8k+1}\right] + \sum_{k=d+1}^{\infty} \frac{16^{n-k}}{8k+1}\right]
\end{aligned}$$

Then, we can use our algorithm for exponentiation modulo $k$ (but hexadecimal instead of binary) to compute this sum.

There are many BBP-type algorithms to calculate mathematical constants. For example, the following polylogarithmic ladders can be used to derive different expressions for $\pi$. Let $w := \frac{1+i}{2}$ and $h := \frac{i}{\sqrt{2}}$. Then, the following are true [Bro98]:

$$\mathrm{Li}_1(w) - \frac{1}{2}\mathrm{Li}_1(\frac{1}{2}) = \frac{\pi i}{4} \tag{4}$$

$$\mathrm{Li}_1(-w^3) - \mathrm{Li}_1(w^2) - \frac{1}{2}\mathrm{Li}_1(\frac{1}{2}) = \frac{\pi i}{4} \tag{5}$$

$$\mathrm{Li}_1(-w^5) - 2\mathrm{Li}_1(w^2) - \frac{1}{2}\mathrm{Li}_1(\frac{1}{2}) = \frac{\pi i}{4} \tag{6}$$

$$\mathrm{Li}_1(h^3) - 2\mathrm{Li}_1(h) - \frac{1}{2}\mathrm{Li}_1(\frac{1}{2}) = \frac{\pi i}{2} \tag{7}$$

They are relatively easy to prove. To prove (4), we can convert the polylogarithm functions into natural logarithms using $\mathrm{Li}_1(z) = -\ln(1-z)$:

$$\begin{aligned}
\mathrm{Li}_1(w) - \frac{1}{2}\mathrm{Li}_1(\frac{1}{2}) &= -\ln(1-w) + \frac{1}{2}\ln(1-\frac{1}{2}) \\
&= -\ln(\frac{1-i}{2}) + \ln\sqrt{2} \\
&= -\ln(\frac{1-i}{\sqrt{2}}) \\
&= -\ln(e^{i(-\frac{\pi}{4})}) \\
&= \frac{\pi i}{4}
\end{aligned}$$

The proofs for the remainder of these polylogarithmic ladders are left as an exercise to the reader.

## References


[Bai06] David Bailey. The bbp algorithm for pi. 10 2006.

[BBP97] David Bailey, Peter Borwein, and Simon Plouffe. On the rapid computation of various polylogarithmic constants. *Mathematics of Computation*, 66(218):903–914, Apr 1997.

[Bro98] D. J. Broadhurst. Polylogarithmic ladders, hypergeometric series and the ten millionth digits of $\zeta(3)$ and $\zeta(5)$, 1998.

[DLMF] *NIST Digital Library of Mathematical Functions.* http://dlmf.nist.gov/, Release 1.0.25 of 2019-12-15. F. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller, B. V. Saunders, H. S. Cohl, and M. A. McClain, eds.

[LL10] Jeffrey C. Lagarias and Wen-Ching Winnie Li. The lerch zeta function i. zeta integrals. *Forum Mathematicum*, 24(1):1–37, Jun 2010.