Symbolic integration with hyperlogarithms

E. Panzer
Institute des Hautes Études Scientifiques Bures-sur-Yvette

erikpanzer@ihes.fr

Multiple polylogarithms

Multiple polylogarithms [1] (abbreviated MPL) are very important and well-understood special functions of several complex variables. They are indexed by a list $n_1, \ldots, n_d \in \mathbb{N}$ of positive integers and may be defined by their power series

$$\operatorname{Li}_{n_1,\ldots,n_d}(z_1,\ldots,z_d) := \sum_{0<k_1<\cdots<k_d} \frac{z_1^{k_1}\cdots z_d^{k_d}}{k_1^{n_1}\cdots k_d^{n_d}} \quad (1)$$

inside the region $|z_d|, |z_d z_{d-1}|, \ldots, |z_d \cdots z_1| < 1$ of absolute convergence. Analytic continuation extends them to multivalued functions like the logarithm $\operatorname{Li}_1(z) = -\log(1-z)$ and the famous dilogarithm $\operatorname{Li}_2(z) = \sum_{n=1}^{\infty} z^n/n^2$ of Euler, which has plenty of interesting properties and applications [2].

These functions fulfil a huge number of functional equations, like for example ($\zeta_2 = \pi^2/6$)

$$\operatorname{Li}_2\left(-z^{-1}\right) = -\zeta_2 - \frac{\log^2(z)}{2} - \operatorname{Li}_2(-z). \quad (2)$$

Also, MPL specialize at $z_1 = \cdots = z_d = 1$ to generalizations of the Riemann zeta values $\operatorname{Li}_n(1) = \zeta(n) = \sum_{k=1}^{\infty} k^{-n}$, the *multiple zeta values* (MZV)

$$\zeta_{n_1,\ldots,n_d} = \sum_{0<k_1<\cdots<k_d} \frac{1}{k_1^{n_1}\cdots k_d^{n_d}} \quad (n_d \geq 2). \quad (3)$$

These fascinating numbers are subject to many conjectures[1], like transcendence of all odd ζ_{2n+1}, and arise from various integrals [4]. For example,

$$\int_{0<t_1<t_2<t_3<1} \frac{\mathrm{d}t_1\mathrm{d}t_2\mathrm{d}t_3}{(1-t_1)t_2(t_3-t_1)} = 2\zeta_3. \quad (4)$$

Apart from applications in mathematics [5], MPL also appear at several places in physics. Most abundantly they emerge from the calculation of some Feynman integrals, which are used to predict quantitatively the interactions of elementary particles. Such calculations are required to compare different theories with actual measurements, as obtained for example in collider experiments at the LHC.

We will present `HyperInt` [6], an open-source program for multiple polylogarithms written in `Maple`. It was designed for the calculation of Feynman integrals in [7, 8], but here we will sketch the underlying ideas and applications to compute transformations like (2) and integrals similar to (4).

Hyperlogarithms

The representation (1) as nested sums opens the door to apply summation algorithms to problems involving polylogarithms, for example see [9].

However, `HyperInt` is a complementary approach and exploits a different structure: polylogarithms are iterated integrals. This means that $\mathrm{d}\operatorname{Li}_1(z) = -\omega_1$ and $\mathrm{d}\operatorname{Li}_{n+1}(z) = \omega_0 \operatorname{Li}_n(z)$ arise from each other by repeated integration with the differential forms $\omega_0 = \mathrm{d}z/z$ and $\omega_1 = \mathrm{d}z/(z-1)$.

More generally, we set $\omega_\sigma := \mathrm{d}z/(z-\sigma)$ for all $\sigma \in \mathbb{C}$ and define for each word $w = \omega_{\sigma_1}\cdots\omega_{\sigma_n}$ the associated *hyperlogarithm* recursively through[2]

$$L_{\omega_\sigma w}(z) := \int_0^z \frac{\mathrm{d}z'}{z'-\sigma} L_w(z') \quad (5)$$

such that $\partial_z L_{\omega_\sigma w}(z) = L_w(z)/(z-\sigma)$. This representation renders the integration and differentiation (with respect to z, for fixed σ) elementary, and we can write any polylogarithm as a hyperlogarithm: $\operatorname{Li}_{n_1,\ldots,n_d}(z_1,\ldots,z_d) = (-1)^d L_w(1)$ if we set $w = \omega_0^{n_d-1}\omega_{\sigma_d}\cdots\omega_0^{n_1-1}\omega_{\sigma_1}$ and $\sigma_k := \prod_{i=k}^{d} z_i^{-1}$.

In this way, linear combinations of words can be used to efficiently represent MPL. Operations on these

[1]Despite lots of efforts, still only very little is known beyond the irrationality [3] of ζ_3.

[2]An exceptional case is the definition $L_{\omega_0^n}(z) := \frac{\log^n(z)}{n!}$, where the integral (5) would diverge.

functions can be rephrased in terms of the words, which is crucial to obtain algorithms suitable for implementation. For example, a product $L_v(z)L_w(z) = L_{v \sqcup\!\sqcup w}(z)$ is a linear combination of hyperlogarithms, where the *shuffle product* $v \sqcup\!\sqcup w$ is the sum of all permutations of vw which preserve the order of the letters of v and w individually; e.g.

$$L_{\omega_\sigma\omega_\tau}L_{\omega_\rho} = L_{\omega_\sigma\omega_\tau\omega_\rho} + L_{\omega_\sigma\omega_\rho\omega_\tau} + L_{\omega_\rho\omega_\sigma\omega_\tau}.$$

Transformation into hyperlogarithms

A given hyperlogarithm $L_w(z)$ can be integrated easily with respect to z, provided that all letters which occur in the word w are independent of z. But often an expression is given in a form where the letters do depend on z. In this case, we must first transform the expression into a representation where all letters are z-independent. As observed in [10], this can be achieved by differentiation under the integral sign and integration by parts, which prove that

$$\mathrm{d}L_w(z) = \sum_{i=1}^{n} L_{\cdots\cancel{\omega_{\sigma_i}}\cdots}(z) \cdot \mathrm{d}\log\frac{\sigma_{i-1}-\sigma_i}{\sigma_i - \sigma_{i+1}} \qquad (6)$$

where we set $\sigma_0 := z$, $\sigma_{n+1} := 0$ while $w = \omega_{\sigma_1}\cdots\omega_{\sigma_n}$ and the hyperlogarithm in the sum has its i'th letter deleted.

This reduction of *weight* (length of the words) yields a recursive algorithm to write any MPL $\mathrm{Li}_{n_1,\ldots,n_d}(z_1,\ldots,z_d)$ or hyperlogarithm $L_{\omega_{\sigma_1}\cdots\omega_{\sigma_n}}(z)$ as a hyperlogarithm in any parameter t, provided that the arguments $z, z_1, \ldots, z_d$ and letters $\sigma_1,\ldots,\sigma_n$ depend only rationally on t.[3]

Example 1 Let $f := \mathrm{Li}_{1,1}(x,y) = L_{\omega_{1/y}\omega_{1/(xy)}}(1)$, we want to rewrite it as a hyperlogarithm in x. The differential (6) with respect to x gives

$$\partial_x f = \frac{L_{\omega_{1/y}}(1)}{x-1} + \left(\frac{1}{x} - \frac{1}{x-1}\right) L_{\omega_{1/(xy)}}(1).$$

Note that for any $\sigma \neq 0$, $L_{\omega_\sigma}(x) = \log(1 - x/\sigma)$. Therefore $L_{\omega_{1/(xy)}}(1) = L_{\omega_{1/y}}(x)$ and we can integrate the above equation to

$$f = L_{\omega_1}(x)L_{\omega_{1/y}}(y) + L_{\omega_0\omega_{1/y}}(x) - L_{\omega_1\omega_{1/y}}(x).$$

The integration constant is fixed because both sides vanish at $x = 0$.

Constants of integration

The above procedure still lacks an important ingredient: the constants of integration are not always zero and need to be determined exactly.

Example 2 It is easy to check that $\mathrm{Li}_1(-1/z) = L_{\omega_0}(z) - L_{\omega_1}(z)$, and together with

$$\partial_z \mathrm{Li}_2(-1/z) = -\frac{1}{z}\mathrm{Li}_1(-1/z)$$

we conclude that $\mathrm{Li}_2(-1/z)$ and

$$L_{\omega_0\omega_{-1}}(z) - L_{\omega_0\omega_0}(z) = -\mathrm{Li}_2(-z) - \frac{\log^2(z)}{2}$$

can only differ by a constant. Comparison with (2) reveals this number as the non-zero $-\zeta_2$.

Essentially, these constants are the limits at zero, because any $L_w(z)$ vanishes at $z=0$. The only exception are the words $w = \omega_0^n$, which yield the divergent $\log^n(z)/n!$. In general one finds that any hyperlogarithm $f(z)$ can be expanded as

$$f(z) = \sum_{k=0}^{n} \log^k(z) f_k(z), \qquad (7)$$

a polynomial in $\log(z)$ with coefficients $f_k(z)$ that are analytic at $z = 0$ (this fixes the f_k uniquely). The required constants of integration are precisely the values $f_0(0)$, which can be computed by a seperate set of algorithms as explained in [11, 6]. Together with the above recursive transformation procedure, these from the core routines of `HyperInt`.

The `HyperInt` library

We will sketch a small subset of `HyperInt` only and refer to [11, 6] and the accompanying worksheet `Manual.mw` for further details.

The program requires no installation and is simply loaded into a `Maple` session by invoking

```
> read "HyperInt.mpl";
```

The package includes the reduction of multiple zeta values up to weight 12 to the (conjectured) basis of the data mine [12].

Representation of polylogarithms

Internally, polylogarithms are represented as lists

$$f = [[g_1, [w_{1,1},\ldots,w_{1,r_1}]], [g_2, [w_{2,1},\ldots,w_{2,r_1}]], \ldots]$$

of pairs of rational functions g_i and lists of words $w_{i,j}$. These encode the function

$$f = \sum_i g_i \cdot \operatorname*{Reg}_{z\to\infty}\left[L_{w_{i,1}}(z)\cdots L_{w_{i,r_i}}(z)\right],$$

where $\mathrm{Reg}_{z\to\infty}$ is the limit $z \to \infty$ with any logarithmic divergences set to zero.[4] Apparently this representation is far from unique (e.g. we could use the shuffle product to enforce $r_i = 1$), but it turns out that this form makes the implementation particularly simple and rather efficient in practice.

[3]This rationality ensures that each $\partial_t \log(\cdots)$ in (6) is a linear combination of terms $1/(t-\tau)$ (τ independent of t).

[4]Explicitly, we expand the hyperlogarithms as in (7), but now $f_k(z)$ are analytic at infinity and $\mathrm{Reg}_{z\to\infty} f(z) := f_0(\infty)$.

For convenient in- and output, `HyperInt` extends the native function `convert` to transform any of the functions

$$\{\texttt{log}, \texttt{ln}, \texttt{polylog}, \texttt{dilog}, \texttt{Hlog}, \texttt{Mpl}, \texttt{Hpl}\}$$

into hyperlogarithms (5) or multiple polylogarithms (1), using the notations

$$\texttt{Hlog}(z, [\sigma_1, \ldots, \sigma_r]) := L_{\omega_{\sigma_1}\cdots\omega_{\sigma_r}}(z) \quad \text{and}$$
$$\texttt{Mpl}([n_1, \ldots, n_r], [z_1, \ldots, z_r]) := \mathrm{Li}_{n_1,\ldots,n_r}(z_1, \ldots, z_r).$$

For example, the dilogarithm $\mathrm{Li}_2(z)$ is

```
> convert(polylog(2,z), Hlog);
> convert(polylog(2,z), Mpl);
```

$$-\operatorname{Hlog}(1, [0, 1/z])$$
$$\operatorname{Mpl}([2], [z])$$

and $L_{\omega_0\omega_{-1}\omega_0}(z) = \mathrm{Li}_3(-z) - \log(z)\,\mathrm{Li}_2(-z)$ from

```
> convert(Hlog(z,[0,-1,0]), Mpl);
```

$$\operatorname{Mpl}([3], [-z]) - \ln(z)\operatorname{Mpl}([2], [-z])$$

Normal forms from fibrations

Due to functional equations, a general polylogarithm $f(\vec{z})$ has many different representations. One way to get a unique representation is by writing

$$f(\vec{z}) = \sum_i c_i L_{w_{i,1}}(z_1) \cdots L_{w_{i,n}}(z_n) \tag{8}$$

as a linear combination of products of hyperlogarithms of words $w_{i,j}$ with the requirement that the letters in $w_{i,j}$ may only depend on the subsequent variables $z_{i+1}, \ldots, z_n$ and such that the factors c_i are constants (with respect to $\vec{z}$). Its implementation constitutes the essential function

$$\texttt{fibrationBasis}(f, [z_1, \ldots, z_n]),$$

which writes a polylogarithm f in the form (8) with respect to the order $\vec{z} = [z_1, \ldots, z_n]$ of variables.

Example 3 This function obtains functional relations between polylogarithms. For example,

```
> fibrationBasis(polylog(2,1-z), [z]);
> convert(%, Mpl);
```

$$-\operatorname{Hlog}(z, [1, 0]) + \zeta_2$$
$$-\operatorname{Mpl}([2], [z]) + \ln(z)\operatorname{Mpl}([1], [z]) + \zeta_2$$

reproduces the classic identity $\mathrm{Li}_2(1-z) = \zeta_2 - \mathrm{Li}_2(z) - \log z \log(1-z)$. Similarly, we obtain (2) as

```
> fibrationBasis(polylog(2,-1/z), [z]);
> convert(%, Mpl);
```

$$-\zeta_2 + \operatorname{Hlog}(z, [0, -1]) - \operatorname{Hlog}(z, [0, 0])$$
$$-\zeta_2 - \operatorname{Mpl}([2], [-z]) - \tfrac{1}{2}\ln(z)^2$$

The analogous inversion relation for $\mathrm{Li}_5(-z^{-1})$ is

```
> fibrationBasis(polylog(5, -1/z), [z]):
> convert(%, Mpl);
```

$$\tfrac{1}{6}\zeta_2 \ln(z)^3 + \tfrac{1}{120}\ln(z)^5 + \operatorname{Mpl}([5], [-z]) + \tfrac{7}{10}\zeta_2^2 \ln(z)$$

As an example involving multiple variables, the five-term relation of the dilogarithm is recovered as

```
> polylog(2,x*y/(1-x)/(1-y))-polylog(2,x
  /(1-y))-polylog(2,y/(1-x)):
> fibrationBasis(%, [x, y]);
```

$$\operatorname{Hlog}(y, [0, 1]) + \operatorname{Hlog}(x, [0, 1])$$
$$-\operatorname{Hlog}(x, [1])\operatorname{Hlog}(y, [1])$$

For more than one variable, each choice $\vec{z}$ of order defines a different basis and a function may take a much simpler form in one basis than in another. For example, $\mathrm{Li}_{1,2}(y, x) + \mathrm{Li}_{1,2}(1/y, xy)$ is just

```
> Mpl([1,2],[y,x])+Mpl([1,2],[1/y,y*x]):
> fibrationBasis(%, [x,y]);
```

$$\operatorname{Hlog}(x, [0, 1/y, 1]) + \operatorname{Hlog}(x, [0, 1, 1/y])$$

but in another basis it takes the longer form

```
> fibrationBasis(%, [y,x]);
```

$$\operatorname{Hlog}(y, [0, 1, 1/x]) + \operatorname{Hlog}(y, [0, 1/x])\operatorname{Hlog}(x, [1])$$
$$-\operatorname{Hlog}(y, [0, 0, 1/x]) - \operatorname{Hlog}(y, [0, 1])\operatorname{Hlog}(x, [1])$$

But still, any order $\vec{z}$ defines a true basis without relations. In particular this means that $f = 0$ if and only if $\texttt{fibrationBasis}(f, \vec{z})$ returns 0, no matter which order $\vec{z}$ was chosen. This allows for an easy check of functional equations.

`HyperInt` ensures that the generated expression holds for $0 \ll z_1 \ll \cdots \ll z_n$. The analytic continuation may introduce imaginary parts like in

```
> fibrationBasis(log(1+z), [z]);
```

$$-I\pi\delta_z + \operatorname{Hlog}(z, [0])$$

where $\delta_z = \pm 1$ indicates the infinitesimal imaginary part assumed for $z \in \mathbb{H}^{\pm}$. For example, the dilogarithm $\mathrm{Li}_2(z)$ develops an imaginary part for $z > 1$:

```
> fibrationBasis(polylog(2, 1+z), [z]);
```

$$I\pi\delta_z \operatorname{Hlog}(z, [-1]) - \operatorname{Hlog}(z, [-1, 0]) + \zeta_2$$

Series expansions

`HyperInt` extends `series` to expand hyperlogarithms near zero. Other expansions are possible with a transformation. So to consider $\mathrm{Li}_{2,3}(1, -z)$ at large z, we substitute $-1/z$ and expand in $z \to 0$:

```
> Mpl([2,3],[1,-1/z]):
> fibrationBasis(%,[z]):
> series(%,z=0,2);
```

$$-2\zeta_2\zeta_3 - 4\zeta_5 - \tfrac{19}{10}\zeta_2^2 \ln(z) - \zeta_3 \ln(z)^2 - \tfrac{1}{6}\zeta_2 \ln(z)^3$$
$$-\tfrac{1}{120}\ln(z)^5 + \left(7 + \zeta_2 - 3\ln(z) + \tfrac{1}{2}\ln(z)^2\right) z + O(z^2)$$

Integration of hyperlogarithms

The main purpose of `HyperInt` is the function

$$\texttt{hyperInt}(f, [z_1 = a_1..b_1, \ldots, z_r = a_r..b_r])$$
$$:= \int_{a_r}^{b_r} \cdots \left[\int_{a_1}^{b_1} f \, \mathrm{d}z_1\right] \cdots \mathrm{d}z_r \tag{9}$$

which computes integrals if the integrands can be transformed into hyperlogarithms.

Example 4 To compute $\int_0^\infty \frac{\mathrm{Li}_{1,1}(-x/y,-y)}{y(1+y)}\mathrm{d}y$, type

```
> hyperInt(Mpl([1,1],[-x/y,-y])/y/(y+1),
   y=0..infinity):
> fibrationBasis(%, [x]);
```

$$\zeta_2 \,\mathrm{Hlog}\,(x,[1]) + \mathrm{Hlog}\,(x,[1,0,1]) - \mathrm{Hlog}\,(x,[0,0,1])$$

Example 5 The integral (4) from [13] is

```
> hyperInt(1/(1-t1)/(t3-t1)/t2,[t1=0..t2,
  t2=0..t3,t3=0..1]):
> fibrationBasis(%);
```

$$2\zeta_3$$

Divergences
Endpoint divergences as in $\int_0^\infty \frac{\ln z}{1+z}\mathrm{d}z$ are detected:

```
> hyperInt(ln(z)/(1+z), z);
  Error, (in integrationStep)
  Divergence at z = infinity of type
  ln(z)^2
```

Linear reducibility
The method of `HyperInt` for integration requires that at each step, the integrand can be expressed as a hyperlogarithm in the next integration variable. This is a very strong restriction which may depend on the order of integration. Suppose we want to integrate

$$f_0(x,y,z) := \frac{1}{((1+x)^2+y)(y+z^2)}$$

over $x, y \in (0,\infty)$. Integration of x yields a hyperlogarithm with irrational letters $-1 \pm i\sqrt{y}$:

$$\int_0^\infty f_0 \,\mathrm{d}x = \frac{\arctan\sqrt{y}}{\sqrt{y}(y+z^2)}.$$

This cannot be written as a hyperlogarithm in y, so the integration of y is not possible as-is with `HyperInt`. But if we integrate over y first, we obtain only letters which are rational functions of x:

$$\int_0^\infty f_0 \,\mathrm{d}y = 2\frac{\log(1+x)-\log(z)}{(x+1+z)(x+1-z)}.$$

Now this is a hyperlogarithm in x, so `HyperInt` can perform its integration over x with the result

$$\int_0^\infty \mathrm{d}x \int_0^\infty \mathrm{d}y\, f_0 = \frac{L_{\omega_1\omega_0}(z) - L_{\omega_{-1}\omega_0}(z)}{z}.$$

Whenever an order of the integration variables exist such that at each step we have a hyperlogarithm in the integration variable, the original integrand is called *linearly reducible*. Luckily, it is not necessary to try out all permutations of the variables in practice because there are algorithms (*polynomial reduction*) which can often find a suitable order if it exists [10, 14].

References

[1] A. B. Goncharov, *Multiple polylogarithms, cyclotomy and modular complexes*, *Math. Res. Lett.* **5** (1998), no. 4 497–516, [`arXiv:1105.2076`].

[2] D. Zagier, *The Dilogarithm Function*, in *Frontiers in Number Theory, Physics, and Geometry II* (P. Cartier, P. Moussa, B. Julia, and P. Vanhove, eds.), pp. 3–65. Springer Berlin Heidelberg, 2007.

[3] R. Apéry, *Irrationalité de $\zeta(2)$ et $\zeta(3)$*, *Journées arithmétiques (Luminy, 1978)* **61** (1979) 11–13.

[4] F. C. S. Brown, *Irrationality proofs for zeta values, moduli spaces and dinner parties*, *ArXiv e-prints* (Dec., 2014) [`arXiv:1412.6508`].

[5] D. Zagier, *Values of Zeta Functions and Their Applications*, in *First European Congress of Mathematics Paris, July 6–10, 1992* (A. Joseph, F. Mignot, F. Murat, B. Prum, and R. Rentschler, eds.), vol. 120 of *Progress in Mathematics*, pp. 497–512. Birkhäuser Basel, 1994.

[6] E. Panzer, *Algorithms for the symbolic integration of hyperlogarithms with applications to Feynman integrals*, *Computer Physics Communications* **188** (Mar., 2015) 148–166, [`arXiv:1403.3385`].

[7] E. Panzer, *On hyperlogarithms and Feynman integrals with divergences and many scales*, *JHEP* **2014** (Mar., 2014) 71, [`arXiv:1401.4361`].

[8] E. Panzer, *On the analytic computation of massless propagators in dimensional regularization*, *Nucl. Phys. B* **874** (Sept., 2013) 567–593, [`arXiv:1305.2161`].

[9] C. Schneider, *Symbolic summation in difference fields and its application in particle physics*, *CA-Rundbrief* **53** (Oct., 2013) 8–12.

[10] F. C. S. Brown, *The Massless Higher-Loop Two-Point Function*, *Commun. Math. Phys.* **287** (May, 2009) 925–958, [`arXiv:0804.1660`].

[11] E. Panzer, *Feynman integrals and hyperlogarithms*. PhD thesis, Humboldt-Universität zu Berlin, 2014. to appear.

[12] J. Blümlein, D. J. Broadhurst, and J. A. M. Vermaseren, *The Multiple Zeta Value data mine*, *Comput. Phys. Commun.* **181** (Mar., 2010) 582–625, [`arXiv:0907.2557`].

[13] F. C. S. Brown, *Multiple zeta values and periods of moduli spaces $\overline{\mathfrak{M}}_{0,n}(\mathbb{R})$*, *Annales scientifiques de l'École Normale Supérieure* **42** (June, 2009) 371–489, [`math/0606419`].

[14] F. C. S. Brown, *On the periods of some Feynman integrals*, *ArXiv e-prints* (Oct., 2009) [`arXiv:0910.0114`].