The Polylogarithm Function

Manuel Eberl

April 18, 2024

Abstract

This entry provides a definition of the *Polylogarithm function*, commonly denoted as $\text{Li}_s(z)$. Here, z is a complex number and s an integer parameter. This function can be defined by the power series expression $\text{Li}_s(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^s}$ for $|z| < 1$ and analytically extended to the entire complex plane, except for a branch cut on $\mathbb{R}_{\geq 1}$.

Several basic properties are also proven, such as the relationship to the Eulerian polynomials via $\text{Li}_{-k}(z) = z(1-z)^{k-1} A_k(z)$ for $k \geq 0$, the derivative formula $\frac{d}{dz}\text{Li}_s(z) = \frac{1}{z}\text{Li}_{s-1}(z)$, the relation to the "normal" logarithm via $\text{Li}_1(z) = -\ln(1-z)$, and the duplication formula $\text{Li}_s(z) + \text{Li}_s(-z) = 2^{1-s}\text{Li}_s(z^2)$.

Contents

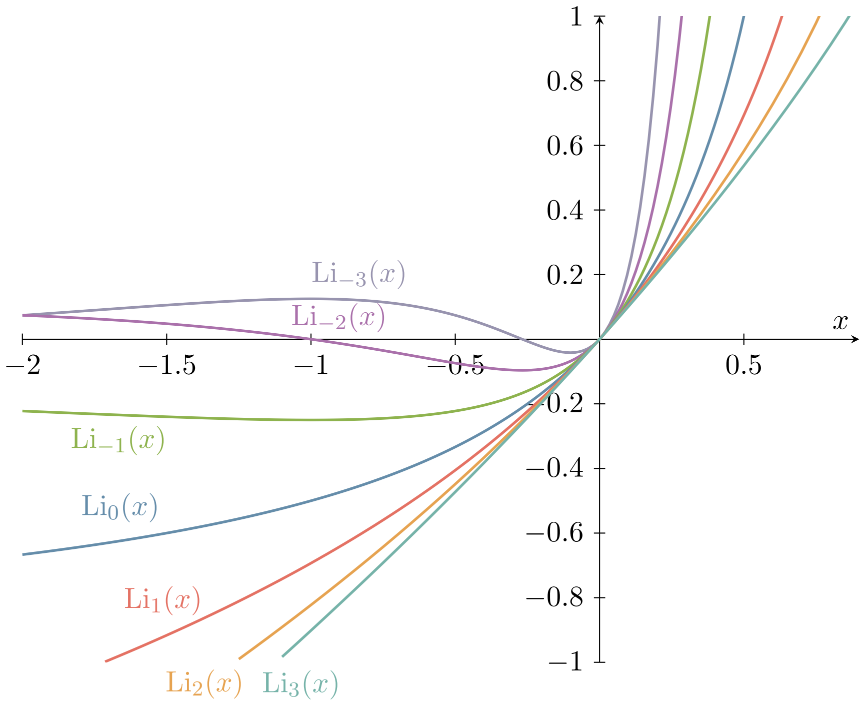

Figure 1: Plots of $\mathrm{Li}_s(x)$ for $s = -3, -2, \ldots, 3$ and real inputs $x \in [-2, 1]$

1 Auxiliary material

theory *Polylog_Library*
imports
  "HOL-Complex_Analysis.Complex_Analysis"
  "Linear_Recurrences.Eulerian_Polynomials"
begin

1.1 Miscellaneous

lemma *fps_conv_radius_fps_of_poly [simp]:*
  fixes *p :: "'a :: {banach, real_normed_div_algebra} poly"*
  shows *"fps_conv_radius (fps_of_poly p) = ∞"*
⟨*proof*⟩

lemma *eval_fps_power:*
  fixes *F :: "'a :: {banach, real_normed_div_algebra, comm_ring_1} fps"*
  assumes *z: "norm z < fps_conv_radius F"*
  shows *"eval_fps (F ^ n) z = eval_fps F z ^ n"*
⟨*proof*⟩

lemma *eval_fps_of_poly [simp]: "eval_fps (fps_of_poly p) z = poly p z"*
⟨*proof*⟩

lemma *poly_holomorphic_on [holomorphic_intros]:*
  assumes *[holomorphic_intros]: "f holomorphic_on A"*
  shows *"(λz. poly p (f z)) holomorphic_on A"*
  ⟨*proof*⟩

lemma *simply_connected_eq_global_primitive:*
  assumes *"simply_connected S" "open S" "f holomorphic_on S"*
  obtains *h* **where** *"⋀z. z ∈ S ⟹ (h has_field_derivative f z) (at z)"*
  ⟨*proof*⟩

lemma
  assumes *"x ∈ closed_segment y z"*
  shows *in_closed_segment_imp_Re_in_closed_segment: "Re x ∈ closed_segment (Re y) (Re z)"* (**is** *?th1*)
    and *in_closed_segment_imp_Im_in_closed_segment: "Im x ∈ closed_segment (Im y) (Im z)"* (**is** *?th2*)
⟨*proof*⟩

lemma *linepath_in_open_segment: "t ∈ {0<..<1} ⟹ x ≠ y ⟹ linepath x y t ∈ open_segment x y"*
  ⟨*proof*⟩

lemma *in_open_segment_imp_Re_in_open_segment:*
  assumes *"x ∈ open_segment y z" "Re y ≠ Re z"*
  shows *"Re x ∈ open_segment (Re y) (Re z)"*
⟨*proof*⟩

lemma *in_open_segment_imp_Im_in_open_segment:*
  assumes *"x ∈ open_segment y z" "Im y ≠ Im z"*
  shows *"Im x ∈ open_segment (Im y) (Im z)"*
⟨*proof*⟩

lemma *poly_eulerian_poly_0 [simp]: "poly (eulerian_poly n) 0 = 1"*
  ⟨*proof*⟩

lemma *eulerian_poly_at_1 [simp]: "poly (eulerian_poly n) 1 = fact n"*
  ⟨*proof*⟩

1.2 The slotted complex plane

lemma *closed_slot_left: "closed (complex_of_real ` {..c})"*
  ⟨*proof*⟩

lemma *closed_slot_right: "closed (complex_of_real ` {c..})"*
  ⟨*proof*⟩

lemma *complex_slot_left_eq: "complex_of_real ` {..c} = {z. Re z ≤ c ∧ Im z = 0}"*
  ⟨*proof*⟩

lemma *complex_slot_right_eq: "complex_of_real ` {c..} = {z. Re z ≥ c ∧ Im z = 0}"*
  ⟨*proof*⟩

lemma *complex_double_slot_eq:*
  "complex_of_real ` ({..c1} ∪ {c2..}) = {z. Im z = 0 ∧ (Re z ≤ c1 ∨ Re z ≥ c2)}"
  ⟨*proof*⟩

lemma *starlike_slotted_complex_plane_left_aux:*
  assumes *z: "z ∈ -(complex_of_real ` {..c})"* **and** *c: "c < c'"*
  shows *"closed_segment (complex_of_real c') z ⊆ -(complex_of_real ` {..c})"*
⟨*proof*⟩

lemma *starlike_slotted_complex_plane_left: "starlike (-(complex_of_real ` {..c}))"*
  ⟨*proof*⟩

lemma *starlike_slotted_complex_plane_right_aux:*
  assumes *z: "z ∈ -(complex_of_real ` {c..})"* **and** *c: "c > c'"*
  shows *"closed_segment (complex_of_real c') z ⊆ -(complex_of_real*

` {c..})"
⟨proof⟩

lemma *starlike_slotted_complex_plane_right: "starlike (-(complex_of_real ` {c..}))"*
⟨proof⟩

lemma *starlike_doubly_slotted_complex_plane_aux:*
assumes *z: "z ∈ -(complex_of_real ` ({..c1} ∪ {c2..}))"* **and** *c: "c1 < c" "c < c2"*
shows *"closed_segment (complex_of_real c) z ⊆ -(complex_of_real ` ({..c1} ∪ {c2..}))"*
⟨proof⟩

lemma *starlike_doubly_slotted_complex_plane:*
assumes *"c1 < c2"*
shows *"starlike (-(complex_of_real ` ({..c1} ∪ {c2..})))"*
⟨proof⟩

lemma *simply_connected_slotted_complex_plane_left:*
"simply_connected (-(complex_of_real ` {..c}))"
⟨proof⟩

lemma *simply_connected_slotted_complex_plane_right:*
"simply_connected (-(complex_of_real ` {c..}))"
⟨proof⟩

lemma *simply_connected_doubly_slotted_complex_plane:*
"c1 < c2 ⟹ simply_connected (-(complex_of_real ` ({..c1} ∪ {c2..})))"
⟨proof⟩

end

2 The Polylogarithm Function

theory *Polylog*
imports
"HOL-Complex_Analysis.Complex_Analysis"
"Linear_Recurrences.Eulerian_Polynomials"
"HOL-Real_Asymp.Real_Asymp"
Polylog_Library
begin

2.1 Definition and basic properties

The principal branch of the Polylogarithm function $\mathrm{Li}_s(z)$ is defined as

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

for $|z| < 1$ and elsewhere by analytic continuation. For integer $s \leq 0$ it is holomorphic except for a pole at $z = 1$. For other values of s it is holomorphic except for a branch cut along the line $[1, \infty)$.

Special values include $\mathrm{Li}_0(z) = \frac{z}{1-z}$ and $\mathrm{Li}_1(z) = -\log(1-z)$.

One could potentially generalise this to arbitrary $s \in \mathbb{C}$, but this makes the analytic continuation somewhat more complicated, so we chosed not to do this at this point.

In the following, we define the principal branch of $\mathrm{Li}_s(z)$ for integer s.

```
definition polylog :: "int ⇒ complex ⇒ complex" where
  "polylog k z =
     (if k ≤ 0 then z * poly (eulerian_poly (nat (-k))) z * (1 - z) powi
(k - 1)
      else if z ∈ of_real ` {1..} then 0
           else (SOME f. f holomorphic_on -of_real`{1..} ∧
                  (∀z∈ball 0 1. f z = (∑n. of_nat (Suc n) powi (-k)
* z ^ Suc n))) z)"

lemma conv_radius_polylog: "conv_radius (λr. of_nat r powi k :: complex)
= 1"
⟨proof⟩

lemma abs_summable_polylog:
  "norm z < 1 ⟹ summable (λr. norm (of_nat r powi k * z ^ r :: complex))"
  ⟨proof⟩
```

Two very central results that characterise the polylogarithm:

$$\mathrm{Li}_s'(z) = \frac{1}{z}\mathrm{Li}_{s-1}(z) \qquad \text{and} \qquad \mathrm{Li}_s(z) = \sum_{n=1}^{\infty} \frac{z^n}{n^s} \quad \text{for } |z| < 1$$

```
theorem has_field_derivative_polylog [derivative_intros]:
        "⋀z. z ∈ (if k ≤ 0 then -{1} else -(of_real ` {1..})) ⟹
              (polylog k has_field_derivative (if z = 0 then 1 else polylog
(k - 1) z / z)) (at z within A)"
  and sums_polylog: "norm z < 1 ⟹ (λn. of_nat (Suc n) powi (-k) * z
^ Suc n) sums polylog k z"
⟨proof⟩

lemma has_field_derivative_polylog' [derivative_intros]:
  assumes "(f has_field_derivative f') (at z within A)"
```

assumes *"if k ≤ 0 then f z ≠ 1 else Im (f z) ≠ 0 ∨ Re (f z) < 1"*
shows *"((λz. polylog k (f z)) has_field_derivative (if f z = 0 then 1 else polylog (k-1) (f z) / f z) * f') (at z within A)"*
⟨*proof*⟩

lemma *polylog_0 [simp]: "polylog k 0 = 0"*
⟨*proof*⟩

A simple consequence of the derivative formula is the following recurrence for Li_s via a contour integral:

$$\mathrm{Li}_s(z) = \int_0^z \frac{1}{w}\mathrm{Li}_{s-1}(w)\,\mathrm{d}w$$

theorem *polylog_has_contour_integral:*
assumes *"z ∉ complex_of_real ` ({..-1} ∪ {1..})"*
shows *"((λw. polylog s w / w) has_contour_integral polylog (s + 1) z) (linepath 0 z)"*
⟨*proof*⟩

lemma *sums_polylog':*
*"norm z < 1 ⟹ k ≠ 0 ⟹ (λn. of_nat n powi - k * z ^ n) sums polylog k z"*
⟨*proof*⟩

lemma *polylog_altdef1:*
*"norm z < 1 ⟹ polylog k z = (∑n. of_nat (Suc n) powi -k * z ^ Suc n)"*
⟨*proof*⟩

lemma *polylog_altdef2:*
*"norm z < 1 ⟹ k ≠ 0 ⟹ polylog k z = (∑n. of_nat n powi -k * z ^ n)"*
⟨*proof*⟩

lemma *polylog_at_pole: "polylog k 1 = 0"*
⟨*proof*⟩

lemma *polylog_at_branch_cut: "x ≥ 1 ⟹ k > 0 ⟹ polylog k (of_real x) = 0"*
⟨*proof*⟩

lemma *holomorphic_on_polylog [holomorphic_intros]:*
assumes *"A ⊆ (if k ≤ 0 then -{1} else -of_real ` {1..})"*
shows *"polylog k holomorphic_on A"*
⟨*proof*⟩

lemmas *holomorphic_on_polylog' [holomorphic_intros] =*

```
  holomorphic_on_compose_gen [OF _ holomorphic_on_polylog[OF order.refl],
unfolded o_def]
```

lemma *analytic_on_polylog [analytic_intros]:*
  assumes *"A ⊆ (if k ≤ 0 then -{1} else -of_real ` {1..})"*
  shows *"polylog k analytic_on A"*
⟨*proof*⟩

lemmas *analytic_on_polylog' [analytic_intros] =*
  analytic_on_compose_gen [OF _ analytic_on_polylog[OF order.refl], unfolded o_def]

lemma *continuous_on_polylog [analytic_intros]:*
  assumes *"A ⊆ (if k ≤ 0 then -{1} else -of_real ` {1..})"*
  shows *"continuous_on A (polylog k)"*
⟨*proof*⟩

lemmas *continuous_on_polylog' [continuous_intros] =*
  continuous_on_compose2 [OF continuous_on_polylog [OF order.refl]]

2.2 Special values

lemma *polylog_neg_int_left:*
  *"k < 0 ⟹ polylog k z = z * poly (eulerian_poly (nat (-k))) z * (1 - z) powi (k - 1)"*
  ⟨*proof*⟩

lemma *polylog_0_left: "polylog 0 z = z / (1 - z)"*
  ⟨*proof*⟩

lemma *polylog_neg1_left: "polylog (-1) x = x / (1 - x) ^ 2"*
  ⟨*proof*⟩

lemma *polylog_neg2_left: "polylog (-2) x = x * (1 + x) / (1 - x) ^ 3"*
  ⟨*proof*⟩

lemma *polylog_neg3_left: "polylog (-3) x = x * (1 + 4 * x + x^2) / (1 - x) ^ 4"*
  ⟨*proof*⟩

lemma *polylog_1:*
  assumes *"z ∉ of_real ` {1..}"*
  shows *"polylog 1 z = -ln (1 - z)"*
⟨*proof*⟩

lemma *is_pole_polylog_1:*
  assumes *"k ≤ 0"*
  shows *"is_pole (polylog k) 1"*
⟨*proof*⟩

```
lemma zorder_polylog_1:
  assumes "k ≤ 0"
  shows   "zorder (polylog k) 1 = k - 1"
⟨proof⟩

lemma isolated_singularity_polylog_1:
  assumes "k ≤ 0"
  shows   "isolated_singularity_at (polylog k) 1"
  ⟨proof⟩

lemma not_essential_polylog_1:
  assumes "k ≤ 0"
  shows   "not_essential (polylog k) 1"
  ⟨proof⟩

lemma polylog_meromorphic_on [meromorphic_intros]:
  assumes "k ≤ 0"
  shows   "polylog k meromorphic_on {1}"
  ⟨proof⟩
```

2.3 Duplication formula

Lastly, we prove the following duplication formula that the polylogarithm satisfies:

$$\mathrm{Li}_s(z) + \mathrm{Li}_s(-z) = 2^{1-s}\mathrm{Li}_s(z^2)$$

The proof is a relatively simple manipulation of infinite sum that defines $\mathrm{Li}_s(z)$ for $|z| < 1$, followed by analytic continuation to its full domain.

```
theorem polylog_duplication:
  assumes "if s ≤ 0 then z ∉ {-1, 1} else z ∉ complex_of_real ` ({..-1}
∪ {1..})"
  shows   "polylog s z + polylog s (-z) = 2 powi (1 - s) * polylog s (z²)"
⟨proof⟩

end
```

References

[1] J. Mason and D. Handscomb. *Chebyshev Polynomials*. CRC Press, 2002.