Polylogarithms in Julia

Prof Matthew Roughan, FIEEE, FACM
Director TRC, Uni Adelaide
<matthew.roughan@adelaide.edu.au>

THE UNIVERSITY of ADELAIDE

TELETRAFFIC RESEARCH CENTRE

Why?

Julia is the cool new kid on the programming block

- It’s fast
- It’s clean

So I should know it, but you need to do, to know

- I wanted a task
- Shouldn’t be easy
- Should make a contribution

Polylogarithm function

$$Li_s(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^s}$$

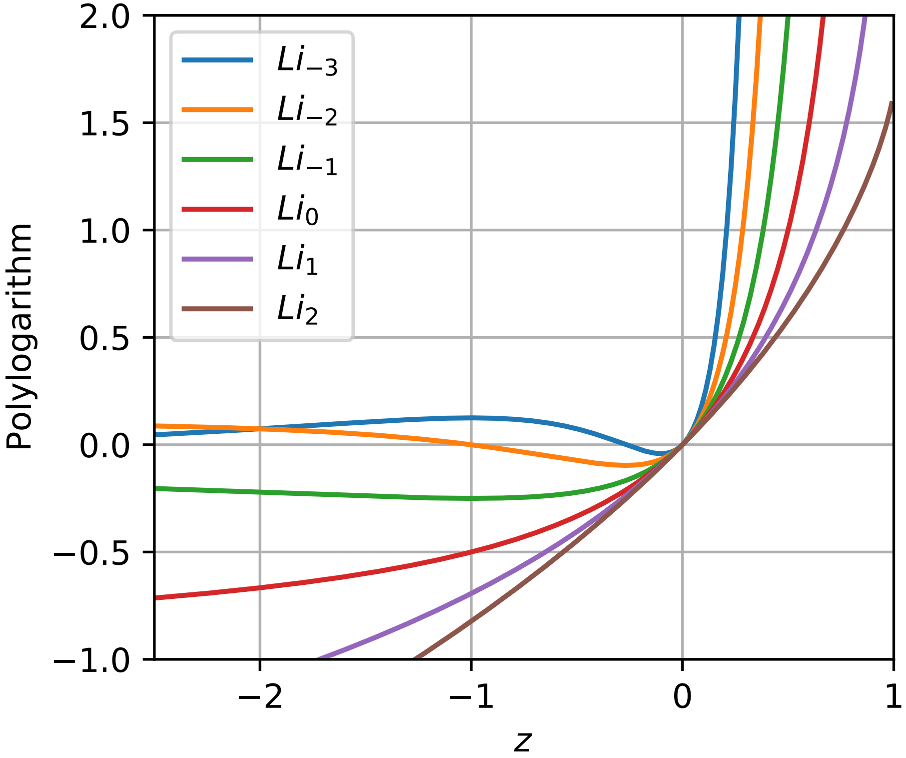
- Converges

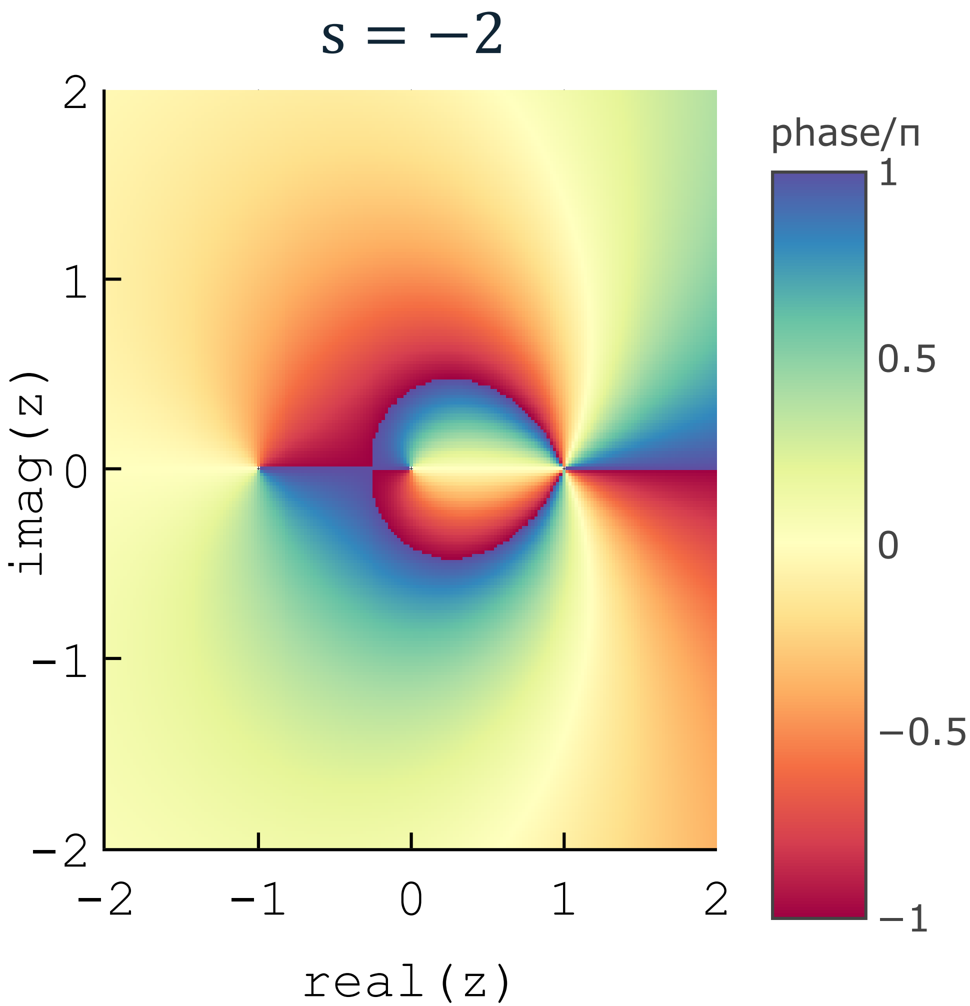
  $|z| < 1$ or $|z| \leq 1$ and $Re(s) \geq 2$

- Analytic continuation to complex plane
  - Except pole at $z = 1$ for $Re(s) < 2$

Polylogarithms

Integer s, real z





Polylogarithm Why?

"almost all the formulae relating to it, have something of the fantastical in them, as if this function alone among all others possessed a sense of humor." Zagier, 2007

Lots of uses and relationships

- Generalisation of log (s=1)
- Riemann zeta function, …
- Fermi-Dirac integrals, …
- Probability
  - Moment generation function of zeta distribution
  - Part of probability mass fn for "Good" distribution
  - Moments of exponential-logarithm distribution

Polylogarithm Calculations

- Should be simple, but actually it's a big mess
  - No-one explains it all
  - Lots of errors in the "literature"
    - Most is unpublished
- Almost no code available
  - Arbitrary precision code (Mathematica)
    - slow
    - proprietary
  - Code for s=2,3, or real z, or some other restricted domain
    - Python: integer s (or non-integral s for |z|<1)
    - Matlab: integer s (mostly, but seems some exceptions)
    - R: integer s>-4, z real, |z|<1

So I Wrote a Package

`https://github.com/mroughan/Polylogarithms.jl`

The easy bit is the calculation
    once you correct all the errors

The hard bit is working out what to do where, and how much

Polylogarithm Calculations

1. Series around $z = 0$

$$Li_s(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^s} \qquad |z| < 1$$

2. Series around $z = 1$

$$Li_s(z) = \Gamma(1-s)(-\ln z)^{s-1} + \sum_{k=0}^{\infty} \frac{\zeta(s-k)}{k!}(\ln z)^k$$

$$|\ln z| < 2\pi$$

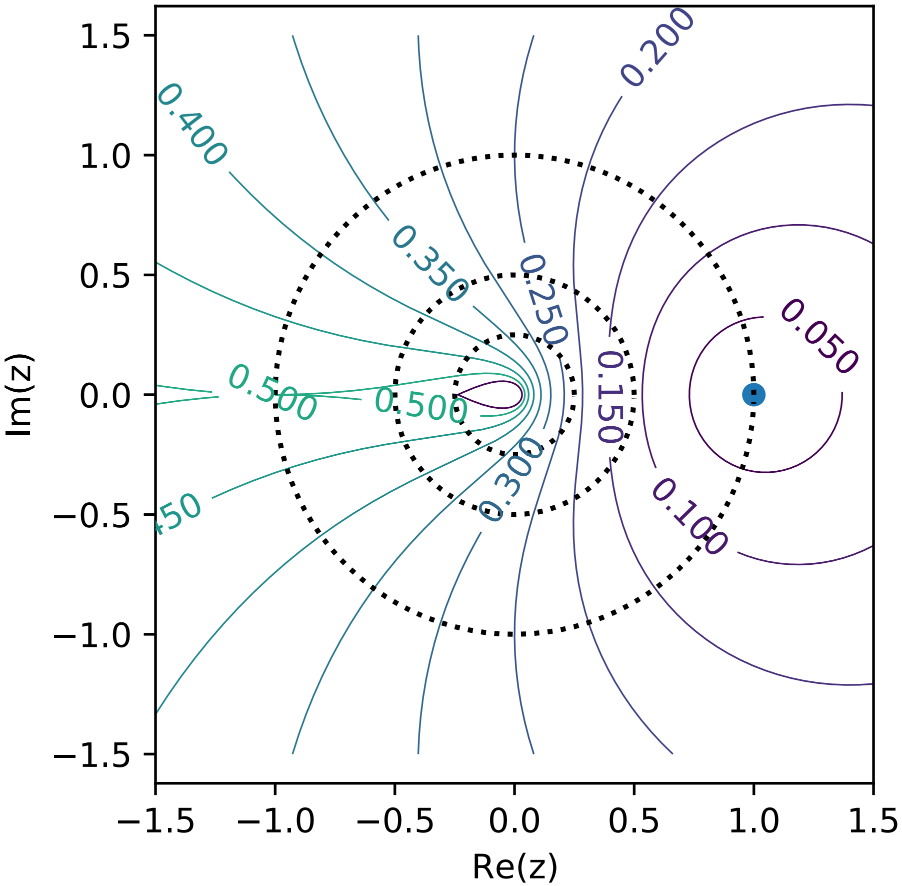
3. Series around $s = n > 0$

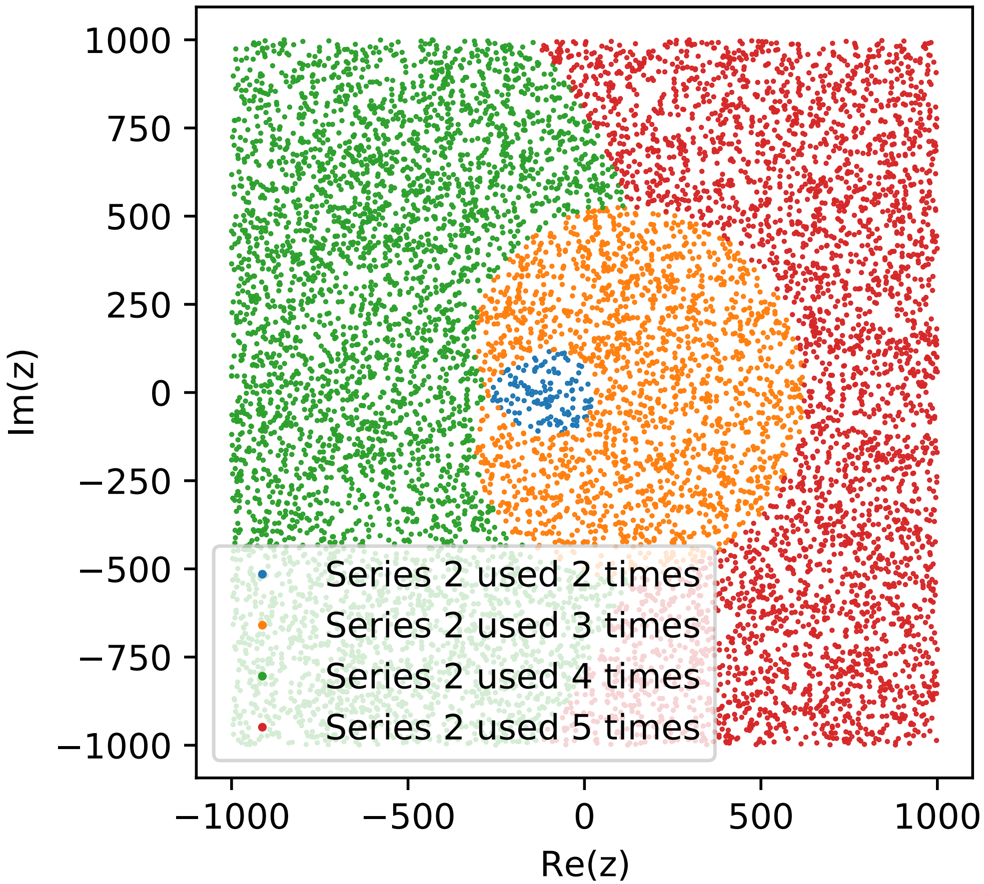
$$Li_{s+\tau}(z) = \frac{(\ln z)^{n-1}}{(n-1)!} Q_{n-1}(L,\tau) + \sum_{k=0}^{\infty} \frac{\zeta(n+\tau-k)}{k!}(\ln z)^k$$

4. Duplication formula

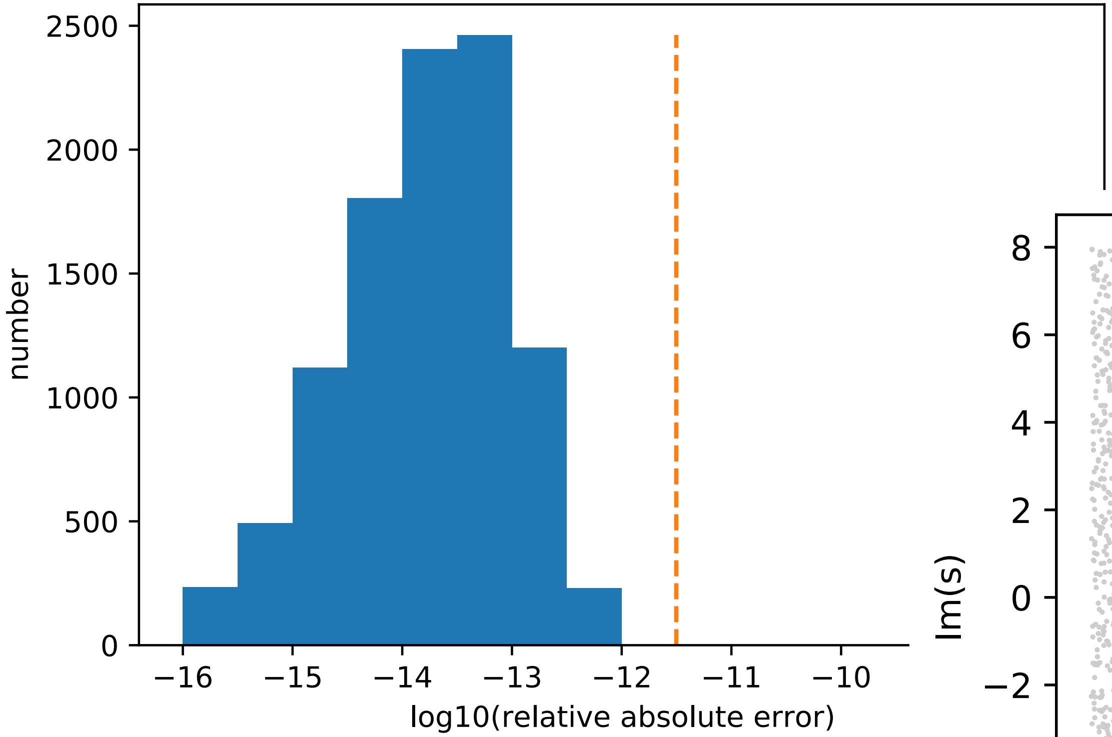
$$Li_s(z) = 2^{s-1}\left[Li_s(\sqrt{z}) + Li_s(-\sqrt{z})\right]$$

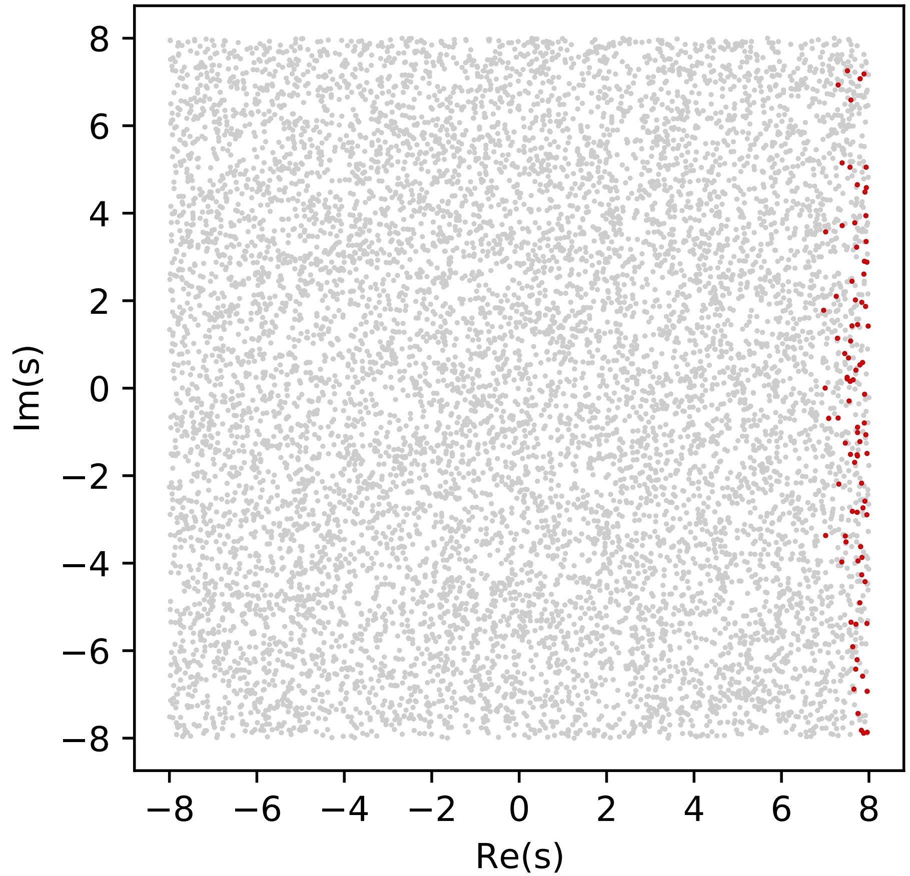
Choosing the right sequence





Some results: accuracy





Some results: performance

Dataset*	Julia	Mathematica	$\zeta(s)$
[-1,1]	30.3 μs	1606.9 μs	1.0 μs
[-8,8]	41.3 μs	1790.0 μs	1.0 μs
[-1000,1000]	143.2 μs	1890.0 μs	0.8 μs

* = z drawn from a square in complex plane

Times measured on Intel i9-10900K CPU running in Julia v1.4.2 with v0.10.3 of the SpecialFunctions package, using a single core, running under Linux Mint 19.3.

https://github.com/mroughan/Polylogarithms.jl

Conclusion

It works

But not 100%

- Failures for Re(s)>8 and large z
- Probably bad cancellation
- There’s another formula to try – didn’t work the first time but improvement in the underlying Julia package might help

https://arxiv.org/pdf/2010.09860.pdf

Extras: I didn't tell you everything

Stopping criteria

- Sequence needs to be decreasing (in magnitude)
- Relative size of summation term $\leq 0.5\ a$. (a=1.0e-12)

Extra functions

$$Q_n(L,\tau) = c_{n,0}(L) + \tau\ c_{n,1}(L) + \ldots$$

where $L = \ln(-\ln(z))$

c is complicated and recursive, but only need a few terms

$$c_{n,0} = H_n - L$$

$$c_{n,1} = -\gamma_1 - \frac{(\psi(n+1)-L)^2}{2} - \left(\frac{\pi^2}{6} - \frac{\psi^{\{(1)\}}(n+1)}{2}\right)$$

$$c_{n,2} = complicated\ \ldots$$

Speed is important

I wanted a job that would be painful in Matlab