GSoC 2025: Special Functions in Julia

Ahmed Y. Kadah[1]
Oscar D. Smith[2]
[1] Student, University of Science and Technology in Zewail City
2 Mentor, [affiliation]

Abstract

In this project, we will work on implementing high speed and high accuracy implementations of various special functions, as well as optimizing and improving existing implementations in Julia. Our work will mostly be on the SpecialFunctions.jl Julia library but will touch on some of the issues in the HyperGeometricFunctions.jl library as well.

Introduction

About me

Hi there! I am Ahmed Yasser Kadah, a third year Communications and Information Engineering student at the University of Science and Technology in Zewail City. TBF.

Background on Special Functions

Special functions are a class of mathematical functions that arise in the solutions of complex physical, engineering, and mathematical problems. These functions are typically defined by specific properties or satisfy particular differential equations, and they include some of the most well-known functions in mathematics, such as the error function, Gamma function, and Bessel functions.

This class of functions serve a unique purpose in many different applications, as they have properties which can be utilized to solve highly complex problems which otherwise have no analytic solutions, or even assist in numerical computation.

Due to the nature and importance of this topic, there already exists a large number of implementations in other languages such as Wolfram Mathematica, MATLAB, and many more. However, the number of memory efficient and open source implementations are somewhat limited to languages like C. Hence, our aim here is not to reinvent the wheel, but refine it.

SpecialFunctions.jl and HyperGeometricFunctions.jl are Julia libraries which aim to provide Julia implementations/interfaces for special functions. While HyperGeometricFunctions.jl is purely implemented in julia, SpecialFunctions.jl makes use of external libraries like openlibm and openspecfun which adds a layer of overhead and reduces the efficiency.

Goals

The goals of the project are to (1) implement some of the functions which currently use external library calls in Julia, (2) work on fixing some of the existing issues in the github repositories, (3) optimizing poorly optimized implementations, (4) explore and add missing functions.

The implementations of functions will make use of different methods of numerical approximation methods like polynomial/rational approximation. Some of the important tools include:

1. Remez.jl, a library for constructing minimax polynomials and rational functions
2. Sollya, a powerful tool

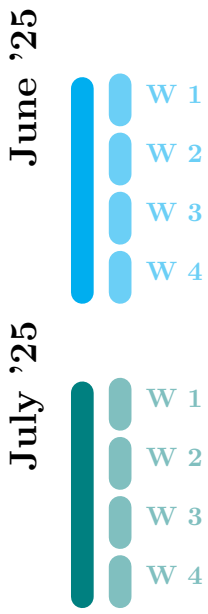
Deliverables and Timeline

Deliverables

- Fully implemented error function and it's varieties
- Implementation of gamma function for real and complex inputs.
- At least 15 issues resolved on SpecialFunctions.jl
- At least 5 issues resolved on HyperGeometricFunctions.jl
- Implementation of Appel Hypergeometric Function

Timeline

TBD



Expected Results

Complete, functional, high efficiency and accuracy, julia implementations of the following:

References

To be organized properly:

1. https://en.wikipedia.org/wiki/Special_functions
2. https://en.wikipedia.org/wiki/Hypergeometric_function
3. https://github.com/JuliaMath/SpecialFunctions.jl
4. https://github.com/JuliaMath/HypergeometricFunctions.jl
5. https://github.com/jishnub/LegendrePolynomials.jl
6. https://reference.wolfram.com/language/guide/SpecialFunctions.html.en
7. https://reference.wolfram.com/language/guide/HypergeometricFunctions.html
8. https://www.mathworks.com/help/matlab/special-functions.html
9. https://www.mathworks.com/help/symbolic/sym.hypergeom.html
10. https://docs.scipy.org/doc/scipy/reference/special.html
11. https://www.gnu.org/software/gsl/