



Google Summer of Code 2021

# Implementing Faddeeva Function

Student: Aman Verma

Mentor: Jonas Eschle



# Overview

Aim: Implement Faddeeva using low level functionalities of Tensorflow(numpy-like) using mathematical algorithms such as Algo 916 and profile the code to find the bottlenecks. Work has been done in scope of zfit.

What is Faddeeva?

The Faddeeva function or Kramp function is a scaled complex complementary error function.

$$w(z) := e^{-z^2} \operatorname{erfc}(-iz) = \operatorname{erfcx}(-iz) = e^{-z^2} \left( 1 + \frac{2i}{\sqrt{\pi}} \int_0^z e^{t^2} dt \right).$$



## Results

We used 3 different Approaches to implement Faddeeva:

1. Using Algorithm 916 + Continued Fraction Expansion

Combined 2 different algorithms for fast and precise calculations. Also gives an option of choosing relerr while calculation which further boosts performance[1.5x-2x] accordingly.

2. Heavily optimized Approach 1 with fixed relerr

Optimized Approach 1 heavily by using Sympy to preprocess some operations and convert loops to vectorized format. Fixed relerr to  $1e-16$ (best precision).

3. Cuda inspired implementation

Here we did some tradeoff between performance and precision and used a simplified version of Faddeeva algorithm. The precision is still good enough( $\sim 1e-9$ ). We beat Scipy implementation by 2x factors for large numbers on GPU.

Input Size	Scipy	Approach 1	Approach 2	Approach 3
1e6	81.4ms	1.23s	747ms	155ms

\*All these values are on GPU



## Major Challenge

TensorFlow is best for doing vectorized computations and does same computations for all elements very efficiently, however the implementations that are around for Faddeeva make heavy use of element-based control flow operations which Tensorflow is not very efficient at optimizing.

## FUN FACT

Although our implementation for faddeeva was not fast enough, we beat Scipy in  $\operatorname{erfcx}(x)$  implementation by 2x-2.5x

## What Next ?

Due to the the nature of the algorithms/implementations around, TensorFlow does not remain a good fit to implement Faddeeva using its lower level functionalities. Even after optimizing series expansions, lookup table, subsidiary functions, vectorizing loops, the performance on GPU reaches near Scipy (best case). Thus, the only way to implement this even faster in TensorFlow would be using kernels as done for Scipy.



Thank You

