Final Project Proposal

Nawat Ngerncham

June 2022

1 Idea Outline

The idea for this project is pretty simple: *parallelized NumPy*.

As everyone who has used NumPy would know, NumPy and the its ND arrays are magical. In addition, its ability to *broadcast* a function to all elements of an ND array, which is very similar to the `map` function in any functional programming language.

However, one drawback that NumPy has is that while it can be programmed such that it runs in parallel by setting some environment variables, it still runs sequentially by default. Thus, the goal of this project is to get as close as possible to and hopefully at match or even beat the performance of vanilla NumPy using Rust and PyO3.

Since NumPy by itself is a very large library(?), we will only implement some of the functions we believe that can be parallelized from for use in linear algebra. Some of these will be from `numpy.linalg` and some will come straight from `numpy`.

2 Measuring Success

We are measuring success using the running time of each function that will implemented compared to vanilla NumPy on large data (suppose a function is for matrix computation, then the matrix would need to be quite large to minimize parallelism overhead. If it can beat NumPy, great. If it cannot, then that's unfortunate.

3 Functions to be Implemented (Tentative)

Please note that some up to all of these may be ignored or has no performance gains over the vanilla NumPy. Please also note that there could be more functions that are implemented that are not part of the following list.

- `numpy.dot`: Dot product of two arrays/vectors
- `numpy.outer`: Outer product of two arrays/vectors
- `numpy.matmul` or `@`: Matrix multiplication of matrices or matrix with arrays/vectors
- `numpy.tensordot`: Tensor dot product
- `numpy.linalg.matrix_power`: Raising a matrix to a power
- `numpy.linalg.norm`: Norm of a vector
- `numpy.linalg.det`: Determinant of matrix
- `numpy.linalg.matrix_rank`: Rank of matrix
- `numpy.linalg.trace`: Trace of matrix
- `numpy.linalg.solve`: Solve a matrix equation
- `numpy.linalg.inv`: Inverse of matrix