# Homework 1: Fast Matrix computation

## 1 General instructions

- Submission format:
  - Your report (containing eventually your solutions, results and comments) in one single PDF file.
  - Your executable code (if you have multiple scripts, put them in a zip).

  For the PDF file, we strongly recommend using LaTeX with a basic template, which you can reuse for each homework assignment.
- This assignment is due on Sunday, January 21st by 23:00:00. NO late submission will be accepted.
- If you have any question related to the homework, feel free to contact TA ZHANG Wenrang via email at 223040237@link.cuhk.edu.cn, or during their office hours (Wednesday 14:00 - 15:00, ZhiXin 4F, seat 72).

## 2 Problem

The objective of this homework is to implement the Strassen's method for matrix multiplication and compare it with classical matrix computation.[1] In order to have a fair comparison, you will implement both the classical and the Strassen's method.

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \times \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}$$

**Standard algorithm**

$$\begin{aligned} h_1 &= a_{1,1}b_{1,1} \\ h_2 &= a_{1,1}b_{1,2} \\ h_3 &= a_{1,2}b_{2,1} \\ h_4 &= a_{1,2}b_{2,2} \\ h_5 &= a_{2,1}b_{1,1} \\ h_6 &= a_{2,1}b_{1,2} \\ h_7 &= a_{2,2}b_{2,1} \\ h_8 &= a_{2,2}b_{2,2} \end{aligned}$$

$$\begin{aligned} c_{1,1} &= h_1 + h_3 \\ c_{1,2} &= h_2 + h_4 \\ c_{2,1} &= h_5 + h_7 \\ c_{2,2} &= h_6 + h_8 \end{aligned}$$

**Strassen's algorithm**

$$\begin{aligned} h_1 &= (a_{1,1} + a_{2,2})(b_{1,1} + b_{2,2}) \\ h_2 &= (a_{2,1} + a_{2,2})b_{1,1} \\ h_3 &= a_{1,1}(b_{1,2} - b_{2,2}) \\ h_4 &= a_{2,2}(b_{2,1} - b_{1,1}) \\ h_5 &= (a_{1,1} + a_{1,2})b_{2,2} \\ h_6 &= (-a_{1,1} + a_{2,1})(b_{1,1} + b_{1,2}) \\ h_7 &= (a_{1,2} - a_{2,2})(b_{2,1} + b_{2,2}) \end{aligned}$$

$$\begin{aligned} c_{1,1} &= h_1 + h_4 - h_5 + h_7 \\ c_{1,2} &= h_3 + h_5 \\ c_{2,1} &= h_2 + h_4 \\ c_{2,2} &= h_1 - h_2 + h_3 + h_6 \end{aligned}$$

[1] The Numpy library in Python provides a multiplication algorithm that is superior to any algorithm you might write, primarily because it addresses memory issues that are NOT covered in this exercise.

**Question 1.** *(20%) Implement two methods for matrix multiplication, specifically for matrices of size* $2^n \times 2^n$*. Test these methods by multiplying matrices* $A = \left(\begin{smallmatrix} 1 & 2 \\ 0 & 1 \end{smallmatrix}\right)$ *and* $B = \left(\begin{smallmatrix} 0 & 1 \\ 3 & 1 \end{smallmatrix}\right)$*. Show on the report your code and its output.*

**Question 2.** *(40%) Execute matrix multiplications for* $N \times N$ *matrices where the entries follow a Gaussian distribution* $\mathcal{N}(0, 1)$ *(use* `numpy.random.rand(N, N)`*) and let* $N = 2^n$ *for* $n = 6, 7, 8, 9$*.*

1. *Record the computation times (use the* `time` *package) and compare them with the theoretical complexity.*
2. *Assess the accuracy by comparing it with numpy's matrix multiplication. Discuss which method (standard method and Strassen's method) is more accurate.*

*Show on the report your results (excluding the matrices themselves) in a clear and understandable format.*

**Question 3.** *(40%) Modify your Strassen's method inspired algorithm to accommodate matrix multiplication for any rectangular matrices. Test your code from the 2 perspectives as in Q2, but with new sizes for the random matrices:* $(50, 60) \times (60, 70)$, $(100, 120) \times (120, 140)$, $(200, 240) \times (240, 280)$, $(300, 360) \times (360, 420)$*. Show on the report your code and your results.*