EPFL

HPC for numerical methods and data analysis

Fall Semester 2024

Prof. Laura Grigori

Assistant: Mariana Martínez Aguilar

Session 11 – November 26, 2024

Deterministic rank revealing factorizations for low rank approximation

Exercise 0 (Optional): Numerical stability of randomized Nystrom

Just as in last week's exercise, consider the MNIST data set and the matrix A as defined in the project or in last week's exercise sheet (with a fixed value of c). **You'll have to be careful with this data set here and in your project because you have to make sure the data set is normalized, i.e. all the entries are between 0 and 1**. Take a relatively small sample of the training set. In this section you can use either last week's code or your code from your project. Consider two different types of sketching matrices, Ω_1, Ω_2 (for example Gaussian and SRHT). For different sketching dimensions of both types of matrices, compute the relative error of the low rank approximation in terms of the nuclear norm. Provide a graph that compares these errors. Comment on your findings.

Exercise 1: Tournament pivoting

The truncated SVD provides the best low rank approximation in terms of the Frobenius and L2 norms. The QR decomposition with column pivoting relies on a permutation matrix Π_c and computes the decomposition $A\Pi_c = QR$, where $A \in \mathbb{R}^{m\times m}, Q \in \mathbb{R}^{m\times n}, R \in \mathbb{R}^{n\times n}$. There are many ways of building such permutational matrix. The strong rank revealing QR, (Strong RRQR) is designed to more effectively identify the numerical rank of a matrix and ensure a robust separation between the significant and insignificant columns in the matrix. For $A \in \mathbb{R}^{m\times n}$ the QR decomposition with column pivoting $A\Pi_c = QR$ can be written as

$$A\Pi_c = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}, \tag{1}$$

where $Q_1 \in \mathbb{R}^{m\times k}$, $Q_2 \in \mathbb{R}^{m\times(n-k)}$, $R_{11} \in \mathbb{R}^{k\times k}, R_{12} \in \mathbb{R}^{k\times(n-k)}, R_{22} \in \mathbb{R}^{(n-k)\times(n-k)}$. Let the singular values of A be ordered as

$$\sigma_{\max}(A) = \sigma_1(A) \geq ... \geq \sigma_{\min}(A) = \sigma_n(A).$$

The factorization in (1) is **strong rank revealing** if for $1 \leq i \leq k$ and $1 \leq j \leq n-k$ we have

$$\begin{aligned} 1 &\leq \frac{\sigma_i(A)}{\sigma_i(R_{11})} \\ \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} &\leq \gamma_1(n,k) \\ \|R_{11}^{-1}R_{12}\|_{\max} &\leq \gamma_2(n,k), \end{aligned}$$

where γ_1, γ_2 are low degree polynomials in n and k.
Recall tournament pivoting for deterministic column selection. (Refer to the image below).

a) Consider a matrix A partitioned into 4 column blocks. Each processor has one of these blocks.

b) *Optional:* Implement strong RRQR. There is a MATLAB implementation, you can use that as a template for building your own Python implementation but if you use ChatGPT to convert MATLAB code into Python be careful of the changes in the index notation.

c) For each block of columns, select k columns using either strong RRQR or column pivoting, save their indices in I_{i0}. Be careful with these indices, you might have to deal with "global" and "local" indices.

d) In each processor output these indices $I_{00}, I_{10}, I_{20}, I_{30}$.

e) Test your method with three different matrices:

- $A = H_n D H_n^\top$, where H_n is the normalized Hadamard matrix of dimension n, D is a diagonal matrix of your choice. Pick n to be "small".
- Load the normalized MNIST data set and build A as in the project (or last week's exercises). Select a few columns and rows.
- Build A from the MNIST data set with 2^{11} data points.

f) Comment your results with the different matrices. Do you notice a problem when you take more rows and columns using the MNIST data set?

g) Using $I_{00}, I_{10}, I_{20}, I_{30}$ build a low rank approximation of A. Check the L2 norm of the error with respect to the error of the truncated SVD.

h) Check if the singular values of these selected columns approximate well the singular values of A.

i) Check if the diagonal elements of R_{11} approximate well the singular values of A.