LRA_introduction

January 4, 2024

1 SVD

1.1 Introduction

Singular value decomposition is similar to eigen value decomposition in the sense that it yields orthonormal bases and values associated with the matrix at hand. It is more genral as it applies to rectangular matrices. For any matrix $A \in \mathbb{R}^{n \times m}$, it reads

$$A = U\Sigma V^T$$

where $U \in \mathbb{R}^{n \times n}$, Σ is a diagonal matrix of the same size of A and $V \in \mathbb{R}^{m \times m}$, Σ. U and V are orthonormal matrices which columns are bases of the spaces associated with each dimension of A.

1.1.1 SVD based approximation

By construction (see literature), SVD provides the best rank k approximation of matrix A in the sense of L^2 norm. Indeed, the singular values σ_i are sorted in a decreasing order i.e. $\forall j > i, \sigma_i > \sigma_j$. Then, we have

$$A \approx A_k = \sum_{i=0}^{k} \sigma_i u_i \otimes v_i$$

1.1.2 SVD is related to EVD

Given the properties of the SVD, we can see that it is possible to solve an EVD problem that is equivalent to solve SVD. Indeed,

$$AA^T = U\Sigma V^T V \Sigma U^T = U\Sigma^2 U^T$$

which means U and Σ^2 are the solution of the EVD of AA^T.

The same can be done with $A^T A = V\Sigma^2 V^T$. Then one can take advantage of the size of the problem if $n >> m$ or the opposite. This also allows one to use the so called power iteration method to compute only the first few modes of the SVD.

1.1.3 Computing SVD

There are multiple algorithms to cumpute SVD of both dense and sparse matrices. Most languages compatible with scientific computing (python, mathlab, C++, fotran,…) have highly efficient libraries to compute SVD on a single machine. Used well, they can use multi-core units and GPUs when available.

Alternatively, one will be able to use EVD solvers when the problem is suitable.

In case of very large matrices, one will have to use more advanced method to solve the problem.

1.2 Exercice

With the programming language of your choice (python/matlab recommended), solve the following exercice.

1. For a function that can be either $f(x, y) = xy$ or $f(x, y) = \sin(\sqrt{x^2 + y^2})$, create a 50 by 200 sampling grid on the unit domain $[0, 1] \times [0, 1]$ and fill a matrix with the values. Then visualize the matrix using the software of your choice.
2. Apply direct SVD from a computing library compatible with (or embedded in) your language. You will have to find out what's the basic computing library used, which algorithm is called and briefly describe its algorithm with pros and cons.
3. Compute approximation error (L2) defined as $E(A, A_k) = \frac{||A-A_k||_{L^2}}{||A||_{L^2}}$ with various truncation rank.
4. Plot the error vs k graph
5. Redo the exercice with a large image of your choice after converting it to grayscale.
6. Add a timer to the SVD call and compare the algorithm based on EVD.