Assignment 2 –

NUMB11/FMNN01: Numerical Linear Algebra

C. Führer/P. Birken/R. Klöfkorn , Numerical Analysis

The purpose of these exercises is to study numerically different methods for QR factorization. Furthermore we study how to apply SVD for compressing an image.

Task 1

Write a program which applies Gram–Schmidt orthogonalization to a given $m \times n$ matrix A ($m \geq n$). It should return n orthogonal basis vectors of range(A). (Note to Python users: Write a class `Orthogonalization` which takes an $m \times n$ array for instantiation. Give this class a method `gramschmidt`. Later this class will get more methods.)

Task 2

Choose some random matrices and test your code. Measure the orthogonality of your matrix by a couple of different criteria:

- Is the 2-norm one?
- How big is the deviation of $Q^{\mathrm{T}}Q$ from the identity matrix? Measure this in the 2-norm.
- Compute the eigenvaluesof $Q^{\mathrm{T}}Q$. What do you expect they should be for an orthogonal matrix?
- Compute the determinant

Make your tests with matrices A with increasing dimensions, e.g. $m = n + 2$ and $n = 1, 10, 100, 1000, 10000$. Report your result. (Python users can define all these tests by methods of the above mentioned class. Note even the command `numpy.allclose`).

Task 3

Use MATLAB's (or scipy.linalg's) command `qr` to compute an orthogonal basis of range(A). Repeat the tests of Task 2. Is there a qualitative difference? What is meant by "Gram-Schmidt is unstable"?

Task 4

Write a MATLAB or Python program performing QR factorization of a full-rank $m \times n$ matrix with Householder transformations. (Python users might want to do this by additing a corresponding method to the above mentioned `class Orthogonalisation`). Test your code by

- checking if Q is indeed orthogonal and $A = QR$.
- comparing your result to MATLAB's qr or Python's scipy . linalg . qr.

Task 5

(cf. Exercise 10.4 in the course book). In the course we considered reflections to orthogonalize a matrix. This led to Householder transformations to perform a QR-factorization. An alternative is to use rotations for the same purpose. The corresponding algorithm is due to Wallace Givens and called *Givens Rotations*.

1. Repeat the chapter on rotations from your linear algebra course. How is a rotation in $\mathbb{R}^n$ described?
2. Repeat the geometrical motivation of Householder transformations. Recall, the goal is to make elements in a certain column to zero to obtain at the end a triangular matrix R.
3. Without looking into literature, try to "invent" a method which uses rotations instead of reflections.
4. If you did not succeeded it is time to consult literature.
5. Describe "your" method or the method you found in the literature by mathematical terms. If you consulted literature, give the corresponding reference in your report.
6. Write a program and verify the method. (Python users do this by adding another method to the previously designed class for orthogonalization methods.)
7. Compare Householder's and Givens' method from the computational effort aspect.

Task 6

In this assignment, you will use the SVD to compress an image. Here are some Python tools, which you might want to use before you start. Matlab users find a more straightforward solution.

- To import an image in Python use

```
from matplotlib.image import imread
```

- This results in case of a colored RGB image in a NumPy array with three indexes (a 3 tensor)
- To convert this image to a grayscale image follow the instructions here:
  https://www.kite.com/python/answers/how-to-convert-an-image-from-rgb-to-grayscale-in-python.
- You obtain now an NumPy array with two indexes (a matrix).

1. Consider an image of a three-striped flag, e.g. the one of France, Germany or Italy and a rank r approximation of it. What rank would you suggest to use?
2. Download the picture kvinna.jpg from the webpage and load it into a suitable python data structure.
3. Perform an SVD of the resulting matrix. As a first test, compute the 2-norm of the difference of the original matrix and the product of the matrices forming the SVD.
4. Now compress the image by neglecting all singular values below a tolerance.
5. Reassemble the matrix and visualize the resulting image
6. How is the quality of the compressed image and how is it impacted by the tolerance?