Winter Term
2023/2024

Prof. Dr. U. Rüde
Benjamin Mann

Algorithms of Numerical Linear Algebra
Assignment 2

Exercise 1 (*Singularity of A^*A*) **2P.**

Given $A \in \mathbb{C}^{m\times n}$ with $m \geq n$, show that A^*A is nonsingular if and only if A has full rank.

Exercise 2 (*QR by Hand*) **3P.**

Using any method you like, determine on paper a reduced QR factorization $A = \hat{Q}\hat{R}$ and a full QR factorization $A = QR$ where

$$A = \begin{pmatrix} 1 & 2 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Exercise 3 (*Givens Rotations*) **4P.**

Consider the orthogonal matrices $F, J \in \mathbb{R}^{m\times m}$

$$F_\theta = \begin{pmatrix} -c & s \\ s & c \end{pmatrix} \quad J_\theta = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$$

where $s = \sin\theta$ and $c = \cos\theta$ for some $\theta \in \mathbb{R}$. The first matrix has $\det F = -1$ and is a reflector — the special case of a Householder reflector in dimension 2. The second has $\det J = 1$ and effects a rotation instead of a reflection. Such a matrix is called a *Givens Rotation*.

(a) Describe exactly what geometric effects left-multiplications by F and J have on the plane $\mathbb{R}^2$. (J rotates the plane by the angle θ, for example, but is the rotation clockwise or counterclockwise?)

(b) Describe an algorithm for QR factorization that is analogous to a Householder QR Factorization but based on Givens rotations instead of Householder reflections.

Exercise 4 (*Least Squares*) **4P.**

Given $A \in \mathbb{C}^{m\times n}$ of rank n and $b \in \mathbb{C}^m$, consider the block 2×2 system of equations

$$\begin{pmatrix} I & A \\ A^* & 0 \end{pmatrix} \begin{pmatrix} r \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix},$$

where I is the $m \times m$ identity. Show that this system has a unique solution $(r, x)^T$, and that the vectors r and x are the residual and the solution of the least squares problem [1, (18.1)].

Exercise 5 (*Reverse Engineering*) **2P.**

```
import numpy as np

def magic(A):
   U,S,V = np.linalg.svd(A)
   eps = np.spacing(1)
   tol = max(np.shape(A))*S[0]*eps
   r = sum(S>tol)
   S = np.diag(np.ones(r)/S[0:r])
   X = np.dot(V[:,0:r], np.dot(S, U[:,0:r].T))
   return X
```

What does the function *magic* compute?

Exercise 6 (*Householder QR Factorization*) **5P.**

Make sure to follow the requirements for programming tasks stated on the information sheet!

(a) Write a Python function `W, R = implicit_qr(A)` that computes an implicit representation of a **full** QR factorization $A = QR$ using Householder reflections. The input is a matrix $A \in \mathbb{C}^{m \times n}$ with $m \geq n$ and full rank n. The output variables are a lower-triangular matrix $W \in \mathbb{C}^{m \times n}$ whose columns are the vectors v_k defining the successive Householder reflections, and a triangular matrix $R \in \mathbb{C}^{m \times n}$.

(b) Write a Python function `Q = form_q(W)` which retrieves the matrix Q from the Householder reflectors. The input value is the matrix W obtained from calling `implicit_qr`. The output is the corresponding unitary matrix $Q \in \mathbb{C}^{m \times m}$.

Note: The function `np.sign` does **not** compute the sign of a complex number as defined in the book.

Hint: The sign of a complex (or real) number z should satisfy $z = \text{sign}(z)|z|$.

References

[1] L.N. Trefethen and D. Bau. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, 1997.