Winter Term
2023/2024

Prof. Dr. U. Rüde
Benjamin Mann

Algorithms of Numerical Linear Algebra
Assignment 3

Exercise 1 (*Big O Notation*) **5P.**

Are the following statements true or false? If a statement is true specify a constant C and a threshold t_0 fulfilling the definitions of the Big O notation given in the lecture.

(a) $\sin x = O(1)$ as $x \to \infty$.

(b) $\sin x = O(1)$ as $x \to 0$.

(c) $\ln x = O(x^{\frac{1}{100}})$ as $x \to \infty$.

(d) $n! = O((\frac{n}{e})^n)$ as $n \to \infty$ (Hint: Stirling's approximation).

(e) $\mathrm{fl}(\pi) - \pi = O(\epsilon_m)$ as $\epsilon_m \to 0$.

(f) $\mathrm{fl}(n\pi) - n\pi = O(\epsilon_m)$ as $\epsilon_m \to 0$ *uniformly* for all integers n.

Exercise 2 (*Stability*) **6P.**

In each of the following tasks, there is a mathematical problem $f\colon \mathbb{C} \to \mathbb{C}$, as well as a corresponding algorithm $\tilde{f}\colon \mathbb{C} \to \mathbb{C}$ s.th. $\tilde{f}(x) \approx f(x)$. Assuming the fundamental axiom of floating point arithmetic [1, (13.7)] and the axiom characterizing the machine epsilon [1, (13.5)] are satisfied, state whether $\tilde{f}$ is *backward stable*, *accurate*, and/or *stable*. Prove your answers!

(a) $f(x) = 2x$ for $x \in \mathbb{C}$. $\tilde{f}(x) = x \oplus x$.

(b) $f(x) = x^2$ for $x \in \mathbb{C}$. $\tilde{f}(x) = x \otimes x$.

(c) $f(x) = 1$ for $x \in \mathbb{C} \setminus \{0\}$. $\tilde{f}(x) = x \oslash x$.

(d) $f(x) = 0$ for $x \in \mathbb{C}$. $\tilde{f}(x) = x \ominus x$.

Exercise 3 (*Given's Rotations: Operation Count*) **4P.**

Recall the algorithm from the previous assignment sheet which computes a QR-factorization using Given's Rotations.

(a) Determine the number of operations n_{op} this algorithm requires to compute the upper triangular matrix R, s.th. $QR = A$ for $A \in \mathbb{C}^{(m+1)\times m}$.

(b) As above, but now assume that A is upper Hessenberg, i.e., all entries below the first subdiagonal are zero, or, equivalently $a_{ij} = 0$ for $i > j + 1$.

For your results, you may ignore lower order terms. Other than in the above task, though, we are very much interested in the constant factor of the highest order term. That means we expect a result of the form

$$n_{\text{op}}(m) \approx Cg(m)$$

with $C \in \mathbb{R}$ and $g \colon \mathbb{N} \to \mathbb{R}$ such that $\frac{n_{\text{op}}(m)}{g(m)} \to C$ as $m \to \infty$.

Exercise 4 (*Given's Rotations: Implementation*) **5P.**

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

(a) Write a Python function `G, R = givens_qr(H)` that computes an implicit representation of a **full** QR factorization $H = QR$ using Given's rotations. The input is an upper Hessenberg matrix $H \in \mathbb{C}^{m+1\times m}$. The output variables are the upper triangular matrix $R \in \mathbb{C}^{m+1\times m}$ and a matrix $G \in \mathbb{C}^{m\times 2}$ whose rows are the tuples (c, s) defining the successive Given's rotations.

(b) Write a Python function `Q = form_q(G)` which retrieves the matrix Q from the Given's rotations. The input value is the matrix G obtained from calling `givens_qr`. The output is the corresponding unitary matrix $Q \in \mathbb{C}^{m+1\times m+1}$.

References

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