Fall 2022: Numerical Analysis
Assignment 4 (due Nov 11, 2022 at 11:59pm ET)

2 extra credit points will be given for generally cleanly written and reasonably well-organized homework. This includes cleanly plotted and labeled figures (see also rules on the first assignment). Use a legend and different line styles to label multiple graphs in one plot (no colors needed). Do not export figures using raster graphics (.jpg, .png) but use vector graphics (.eps, .pdf, .dxf) that do not mess up lines. Label axes and use titles.

1. **Basic properties of Householder matrices [2+1+1pts]** Let $n \geq 2$ and denote by H and H_1 the Householder matrices corresponding to the nonzero vectors $\boldsymbol{v} \in \mathbb{R}^n$ and $\boldsymbol{v}_1 \in \mathbb{R}^n$, respectively.
   (a) Find the eigenvalues of H. This is easiest if you consider vectors in directon of $\boldsymbol{v}$ and orthogonal to $\boldsymbol{v}$ as candidates for eigenvectors.
   (b) Compute the determinant of H.
   (c) Show that HH_1 is an orthogonal matrix, but it cannot be a Householder matrix for any vector $\boldsymbol{w}$.

2. **Orthogonalization methods [3+3pts]**
   (a) Use Householder matrices to compute the QR-factorization of the matrix
$$\begin{bmatrix} 9 & -6 \\ 12 & -8 \\ 0 & 20 \end{bmatrix}.$$
   Write down both formulations we discussed in class, i.e., $A = \hat{Q}\hat{R}$ with $\hat{Q} \in \mathbb{R}^{m\times n}$, $\hat{R} \in \mathbb{R}^{n\times n}$ as well as $A = QR$ with $Q \in \mathbb{R}^{m\times m}$, $\hat{R} \in \mathbb{R}^{m\times n}$.
   (b) Let us generalize a result from class. Given any two nonzero vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{R}^n$, construct a Householder matrix H, such that $H\boldsymbol{x}$ is a scalar multiple of $\boldsymbol{y}$. Is the matrix H unique?

3. **QR decomposition and linear systems [2+2pts]**.
   (a) Assume you have a QR-factorization of $A \in \mathbb{R}^{m\times n}$, i.e., $A = QR$. Show how this QR factorization can be used to compute $\kappa_2(A^TA)$, where $\kappa_2(\cdot)$ is the condition number with respect to the 2-norm.
   (b) Now assume you have a QR-factorizaton of an invertible, square matrix $A \in \mathbb{R}^{n\times n}$, i.e., $A = QR$. Show how this factorization can be used to solve the linear system $A\boldsymbol{x} = \boldsymbol{b}$, for given $\boldsymbol{b} \in \mathbb{R}^n$. How many floating point operations does this linear system solve require (assume you already know the QR-factorization!) How does that compare with the number of flops needed when you have given an LU-decomposition of A?

4. **Gram-Schmidt versus Householder [1+1+2+2pts]** The QR-decomposition $A = QR$ for $A \in \mathbb{R}^{m\times n}$, $m \geq n$ with orthogonal matrix $Q \in \mathbb{R}^{m\times m}$ and upper triangular matrix $R \in \mathbb{R}^{m\times n}$ can not only be constructed using Householder reflections, but also through Gram-Schmidt orthogonalization, which you probably know from your linear algebra class. Gram Schmidt works as follows: First, we write the matrix $A = [\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(n)}]$, where $\boldsymbol{a}^{(i)} \in \mathbb{R}^m$ are the columns of the matrix A. Then, Gram-Schmidt works as follows:
$$\boldsymbol{q}^{(1)} := \frac{\boldsymbol{a}^{(1)}}{\|\boldsymbol{a}^{(1)}\|_2}, \text{ and for } i = 2, \ldots, n$$

$$\boldsymbol{b}^{(i)} := \boldsymbol{a}^{(i)} - \sum_{k=1}^{i-1} (\boldsymbol{q}^{(k)})^T \boldsymbol{a}^{(k)} \boldsymbol{q}^{(k)}, \quad \boldsymbol{q}^{(i)} := \frac{\boldsymbol{b}^{(i)}}{\|\boldsymbol{b}^{(i)}\|_2}.$$

(a) Show that the matrix $\hat{Q} := [\boldsymbol{q}^{(1)}, \ldots, \boldsymbol{q}^{(n)}]$ is orthogonal.

(b) Use the above algorithm steps to write the $\boldsymbol{a}^{(i+1)}$ as a linear combination of the vectors $\boldsymbol{q}^{(1)}, \ldots, \boldsymbol{q}^{(i+1)}$.

(c) The above formula can be used to write $AR_1 \cdots R_n = \hat{Q}$ with upper triangular matrices R_i. Give these matrices $R_i \in \mathbb{R}^{n\times n}$. Since we know that $\hat{R} := (R_1 \cdots R_n)^{-1}$ is again an upper triangular matrix, we have thus found an alternative method to compute a (reduced) QR decomposition $A = \hat{Q}\hat{R}$ using Gram Schmidt instead of Householder.

(d) Consider

$$A := \begin{bmatrix} 1+\varepsilon & 1 & 1 \\ 1 & 1+\varepsilon & 1 \\ 1 & 1 & 1+\varepsilon \end{bmatrix} \tag{1}$$

with a small positive number ε (e.g., $\varepsilon = 10^{-10}$) such that ε^2 can be ignored in numerical computations (i.e., you drop terms involving ε when they are added to terms of order 1 or order ε, i.e., you round $3 + 2\varepsilon + \varepsilon^2 \approx 3 + 2\varepsilon$). Using this rule, use the Gram Schmidt procedure to compute the matrix Q arising in the QR-factorization of A. For the result, compute the angle between the vectors given by the columns. For an orthogonal matrix, these should all be $\pi/2$. What do you find?[1]

5. **Rounding errors [1+2+3pt]** Computers use finite precision to represent real numbers, which leads to rounding. You can see the size of the rounding error for real numbers around 1 using the MATLAB command `eps(1)` or the numpy command `np.spacing(1)`. This number, also called the *machine epsilon*, is $\epsilon = 2.22 \times 10^{-16} \ldots$ for the standard (double precision) representation of numbers in a computer. Try the following experiments in MATLAB (or Python/Octave/Julia).[2]

(a) Report the analytical/exact result, and the result you get when using your computer for:

$$a = (1-1) + 10^{-16}, \quad b = 1 - (1 + 10^{-16}).$$

What do you think is happening?

(b) The n-th Hilbert matrix $H_n \in \mathbb{R}^{n\times n}$ has the entries $h_{ij} = (i+j-1)^{-1}$ for $i, j = 1, \ldots, n$.[3] It is known that solving systems with the Hilbert matrix increases rounding errors since the matrix is poorly conditioned. Let $\boldsymbol{e}_n$ be the column vector of length n that contains all 1's. Report the exact and the numerically computed values for

$$\|H_n(H_n^{-1}\boldsymbol{e}_n) - \boldsymbol{e}_n\|, \text{ for } n = 5, 10, 20.$$

Here, $\|\cdot\|$ is the usual Euclidean norm. Also report the condition numbers of H_n (with respect to either norm) for $n = 5, 10, 20$.

[1] What you observe is more general, namely that Gram-Schmidt orthogonalization is sensitive to rounding errors. Householder is much more robust as the matrix Q when derived from Householder is always the product of orthogonal matrices, even if there is roundoff to compute the Householder vectors $\boldsymbol{v}$. Thus, Householder is generally preferred over Gram-Schmidt for computing QR-factorizations.

[2] You can use the command `format long` to get 15 digits output from your computer. If you need more digits of a number a, you can use `fprintf('%2.20f\n',a)` to see 20 digits.

[3] You can get H_n by using the MATLAB command `hilb(n)`.

(c) As you know, for differentiable functions f holds

$$f'(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$

Thus, to numerically approximate a derivative of a function for which the derivative is hard to derive analytically, one can use, for a small h, the approximation

$$f'(x_0) \approx f'_{\text{num}}(x_0) := \frac{f(x_0 + h) - f(x_0)}{h}.$$

In this approximation, one subtracts very similar numbers in the numerator of the fraction from each other, and then multiplies with a large number (namely h^{-1}), which can lead to errors that are much larger than the machine epsilon. Compute approximations of the derivative of the function

$$f(x) = \frac{\exp(x)}{\cos(x)^3 + \sin(x)^3}$$

at $x_0 = \pi/4$. In order to do so, use progressively smaller perturbations $h = 10^{-k}$ for $k = 1, \ldots 16$. Present the errors in the resulting approximation in a log-log plot, i.e., use a logarithmic scale to plot the values of h on the x-axis, and a logarithmic scale to plot the errors between the finite difference approximation $f'_{\text{num}}(x_0)$ and the exact value, which is $f'(x_0) = 3.101766393836051$, on the y-axis.[4] What do you observe as h becomes smaller, and for which h do you get the best approximation to the derivative?

6. **Eigenvalue/vector properties, [11pts]** Prove the following statements, using the basic definition of eigenvalues and eigenvectors, or give a counterexample showing the statement is not true. Assume $A \in \mathbb{R}^{n \times n}$, $n \geq 1$.

(a) If λ is an eigenvalue of A and $\alpha \in \mathbb{R}$, then $\lambda + \alpha$ is an eigenvalue of $A + \alpha I$, where I is the identity matrix.

(b) If λ is an eigenvalue of A and $\alpha \in \mathbb{R}$, then $\alpha\lambda$ is an eigenvalue of αA.

(c) If A is invertible and λ is an eigenvalue of A, then $1/\lambda$ is an eigenvalue of A^{-1}.

(d) If λ is an eigenvalue of A, then for any positive integer k, λ^k is an eigenvalue of A^k.

(e) If B is "similar" to A, which means that there is a nonsingular matrix S such that $B = SAS^{-1}$, then if λ is an eigenvalue of A, it is also an eigenvalue of B. How do the eigenvectors of B relate to the eigenvectors of A?

(f) Every matrix with $n \geq 2$ has at least two distinct eigenvalues, say λ and μ, with $\lambda \neq \mu$.

(g) Every real matrix has a real eigenvalue.

(h) If A is singular, then it has an eigenvalue equal to zero.

(i) If all the eigenvalues of a matrix A are zero, then $A = 0$.

(j) If A is symmetric, then all eigenvalues of A are real.

(k) If A is symmetric, then eigenvectors of A corresponding to different eigenvalues are orthogonal.

[4] MATLAB and Python offer a `loglog` function to do that.