Exercise set 5 – MATH-250 Advanced Numerical Analysis I

The exercise sheet is divided into two sections: quiz and exercises. The quiz will be discussed in the beginning of the lecture on Thursday, March 27. The exercises marked with ($\star$) are graded homework. The exercises marked with **(Python)** are implementation based and can be solved in the Jupyter notebooks which are available on Moodle/Noto. **The deadline for submitting your solutions to the homework is Friday, March 28 at 10h15.**

Quiz

(a) Consider a linear system $A\mathbf{x} = \mathbf{b}$ with a given matrix $A \in \mathbb{R}^{n\times n}$ and a right-hand side b. Which of the following statements are correct?

(i) The linear system has a solution if and only if A is invertible (that is, $\det A \neq 0$).

☐ True ☐ False

(ii) If A is not invertible then there is either no solution or infinitely many solutions.

☐ True ☐ False

(iii) A random matrix (that is, a matrix with independent normally distributed entries) is invertible with probability 1.

☐ True ☐ False

(b) What is the complexity of Gaussian elimination for solving $A\mathbf{x} = \mathbf{b}$?

☐ $O(n)$ ☐ $O(n^3)$

☐ $O(n!)$ ☐ $O(n^2)$

(c) Let $\|\cdot\|_p$ denote the ℓ^p norm of a vector for $1 \leq p \leq \infty$. Which of the following statements are correct?

(i) $\|\mathbf{x}\|_1 \leq n\|\mathbf{x}\|_\infty$ and $\|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_1$ for $\mathbf{x} \in \mathbb{R}^n$

☐ True ☐ False

(ii) $\|\mathbf{x}\|_2 \leq \sqrt{n}\|\mathbf{x}\|_1$ and $\|\mathbf{x}\|_1 \leq \|\mathbf{x}\|_2$ for $\mathbf{x} \in \mathbb{R}^n$

☐ True ☐ False

(iii) $|\langle \mathbf{x}, \mathbf{y}\rangle| \leq \|\mathbf{x}\|_p\|\mathbf{y}\|_p$ for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and any $1 \leq p \leq \infty$

☐ True ☐ False

(iv) $\|\cdot\|_p$ is not a norm for $p = 1/2$

☐ True ☐ False

Exercises

Problem 1. The goal of this exercise is to prove Theorem 2.12; the three-term recurrence relation for the Legendre polynomials defined in the lecture notes:

$$(n+1)q_{n+1}(x) = (2n+1)xq_n(x) - nq_{n-1}(x), \qquad |x| < 1. \tag{1}$$

(a) Using that $q_0, \dots, q_n$ is an orthogonal basis for $\mathbb{P}_n$, we consider the expansion

$$xq_n = \sum_{i=0}^{n+1} \alpha_i q_i, \quad \alpha_i = \frac{\langle xq_n, q_i \rangle}{\langle q_i, q_i \rangle},$$

where $\langle \cdot, \cdot \rangle$ denotes the L^2 inner product on $[-1, 1]$. Show that $\alpha_i = 0$ except for $i = n-1, n+1$.

(b) Use Theorem 2.11 to determine the leading coefficient of q_n and use this to show that

$$\frac{2n+1}{n+1} xq_n - q_{n+1} \in \mathbb{P}_n$$

(c) Using (b), compute α_{n-1} and α_{n+1}. You may use $\langle q_n, q_n \rangle = \frac{2}{2n+1}$. From this deduce the recurrence relation (1).

Bonus: Prove $\langle q_n, q_n \rangle = \frac{2}{2n+1}$ using Theorem 2.12.

Problem 2. Using Theorem 2.12, show that any root λ of the Legendre polynomial q_{n+1} is also a root of $\det(\lambda B - A)$ with

$$B = \begin{bmatrix} 1 & & & \\ & 3 & & \\ & & \ddots & \\ & & & 2n+1 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 & & & \\ 1 & 0 & 2 & & \\ & 2 & \ddots & \ddots & \\ & & \ddots & \ddots & n \\ & & & n & 0 \end{bmatrix}$$

Verify this statement for small n (say $n = 1, 2$) in Python with `scipy.linalg.eigvalsh(A,B)`.

Bonus: Prove Theorem 2.15 using this result.

Problem 3. (Python) Consider the logarithmic spiral curve, whose x- and y-coordinate at time t is given by

$$\begin{cases} x(t) = \exp(-at)\cos(t), \\ y(t) = \exp(-at)\sin(t); \end{cases} \qquad t \in [0, 8\pi]. \tag{2}$$

The parameter $a > 0$ controls how rapidly the spiral curves inward.

(a) Write a Python function which approximates the length of the logarithmic spiral (2). Specifically, ask Chat-GPT to give you the formula for computing the length of a general, smooth curve. Approximate the integral in this formula using the simple Gaussian quadrature with $n = 5$ quadrature points, which is implemented in `scipy.integrate.fixed_quad/sp.integrate.fixed_quad`. Use your function to compute the length of the logarithmic spiral (2) for $a = 0.1$ and $a = 0.5$.

(b) Compute the exact length of the logarithmic spiral. Specifically, Ask Chat-GPT to (analytically) compute the length of the logarithmic spiral with the formula it gave you previously. Evaluate the expression for $a = 0.1$ and $a = 0.5$. How far away are they from the approximation?

Warning: It often happens that Chat-GPT gives a wrong answer. Verify that every step in the provided explanation is reasonable, and correct it if necessary.

(c) Visualize the logarithmic spiral for $a = 0.1$ and $a = 0.5$ along with the 5 quadrature nodes of the simple Gaussian quadrature. You can obtain the nodes for the interval $[-1, 1]$ with the function `scipy.special.roots_legendre/sp.special.roots_legendre`, but will have to rescale them to the interval $[0, 8\pi]$. Use their locations to explain why the approximation error is significantly larger for one of the values of a.

($\star$) Problem 4.

We consider the integral

$$\int_{-1}^{1} f(x)w(x)\,\mathrm{d}x, \quad w(x) = \frac{1}{\sqrt{1-x^2}}.$$

Additionally, we define the following inner product

$$\langle u, v \rangle_w = \int_{-1}^{1} u(x)v(x)w(x)\,\mathrm{d}x. \tag{3}$$

(a) Consider

$$I_n = \int_{-1}^{1} x^n w(x)\,\mathrm{d}x.$$

For n odd, meaning that $n \equiv 1 \mod 2$, explain why $I_n = 0$. Compute the value of I_0. For even n, meaning $n \equiv 0 \mod 2$, derive the recurrence relation

$$I_n = \frac{n-1}{n} I_{n-2}.$$

(b) Apply the Gram-Schmidt algorithm to orthogonalize the monomials $1, x, x^2, x^3$ with respect to the inner product $\langle \cdot, \cdot \rangle_w$ as defined in (3). Use the result from (a) to perform these calculations. Normalize the resulting orthogonal polynomials p_0, p_1, p_2, and p_3 such that $p_i(1) = 1$ for $i = 0, 1, 2, 3$.

It turns out that the resulting polynomials, orthogonal with respect to the scalar product defined in (3), are the so called *Chebyshev polynomials*, and it can be shown that the roots of p_{n+1} are the *Chebyshev nodes*

$$x_i = \cos\left(\pi \frac{2i+1}{2(n+1)}\right), \quad i = 0, 1 \ldots, n.$$

Write a Python script that verifies this equality. For the handling of polynomials, NumPy provides the class `numpy.polynomial.polynomial.Polynomial`. Use this class and its associated function `roots` to compute the roots.

Hint: The Jupyter notebook provided on Moodle imports the `Polynomial` class under the alias `poly`. Hence, you can directly call `poly` to create your polynomials.

(c) Now let $p_{n+1} \in \mathbb{P}_{n+1}$ be the polynomial that is orthogonal to $\mathbb{P}_n$ with respect to (3) and satisfies $p_{n+1}(1) = 1$. The polynomial p_{n+1} has $n+1$ distinct roots $x_0, \ldots, x_n \in (-1, 1)$, defined above. Consider the quadrature rule defined by

$$Q_n[f] = \sum_{i=0}^{n} \alpha_i f(x_i), \quad \alpha_i = \int_{-1}^{1} \ell_i(x) w(x) \, \mathrm{d}x,$$

where $\ell_0, \ell_1, \ldots, \ell_n$ are the usual Lagrange polynomials associated with $x_0, x_1, \ldots, x_n$.

Show that the quadrature rule Q_n has order $2n + 2$ for the weighted integral, that is,

$$Q_n[p] = \int_{-1}^{1} p(x) w(x) \, \mathrm{d}x \quad \forall p \in \mathbb{P}_{2n+1}.$$

Hint: Adapt the arguments made in the beginning of Section 2.5 in the lecture notes.

(d) Write a Python function `cheb_quad(f, num)` implementing the quadrature rule $Q_n[f]$ from (c) using the weights

$$\alpha_i = \frac{\pi}{n+1}, \quad i = 1, 2, \ldots, n.$$

Apply $Q_n[f]$ to $f_1(x) = \frac{|x|^{1/5}}{|x+2|+|x-2|}$ and $f_2(x) = \frac{\exp(-x^2)}{\cos x|x|}$ for $n = 1, 2, \ldots, 1000$. Display the approximation errors of f_1 on a loglog plot, and the approximation errors of f_2 on a semilogy plot, with the x-axis showing the number of nodes n. For the computation of the reference integral you can use SciPy's integration module with the function `scipy.integrate.quad(f, -1, 1, epsabs=1e-16)`. Make sure you use the correct function f!

Remember to upload a scan `homework05.pdf` of your solutions and the completed Jupyter notebook `homework05.ipynb` corresponding to the homework to the submission panel on Moodle until Friday, March 28 at 10h15. To download your notebook from Noto, use `File > Download`. Only your submissions to Moodle will be considered for grading.