Numerical Analysis: Midterm (**50 marks**)

Urbain Vaes

Fall 11111100101

You are not required to complete every question.Although the total marks on the exam sum to 55, your midterm grade will be calculated out of 50.

Question 1 (Floating point arithmetic, **10 marks**). True or false?

1. Let $(\bullet)_2$ denote binary representation. Then $(0.1101)_2 + (0.0011)_2 = (1.0)_2$.
2. Let ε_{64} denote the machine epsilon for the `Float64` format, i.e. `eps(Float64)`. Then the number 2 is representable exactly in this format, and the next representable number is $2 + \varepsilon_{64}$.
3. It holds that $(10000)_2 \times (0.1010)_2 = (1010)_2$.
4. In Julia, `Float64(.6) == Float32(.6)` evaluates to `true`.
5. The spacing (in absolute value) between successive single-precision (`Float32`) floating point numbers is constant.
6. Infinitely many distinct real numbers can be represented exactly in the `Float64` format, but only finitely many can be represented exactly in the `Float32` format.
7. It holds that $(0.\overline{101})_2 = \frac{5}{7}$.
8. Machine addition $\widehat{+}$ is an operation that is *associative* but not *commutative*.
9. The machine epsilon is the smallest number of the form 2^{-n} with $n \in \mathbf{N}$ that can be represented exactly in a floating point format.
10. In Julia, the expression `1 + eps()/3 == 1 + eps()` evaluates to `true`.
11. **Bonus.** In Julia, the expression `exp(eps()/2) == 1 + eps()` evaluates to `true`.

    Explain briefly:

12. **Bonus.** In Julia, the expression `cos(eps()) == 1` evaluates to `true`.

    Explain briefly:

Question 2 (Interpolation and approximation, **10 marks**). Are the following statements true or false? Prove or disprove. Recall that $\mathcal{P}_d$ denotes the set of polynomials of degree at most d.

1. Assume that $x_0 < x_1 < x_2 < x_3$ and y_0, y_1, y_2, y_3 are given real numbers. Then there exists a polynomial $p \in \mathcal{P}_3$ such that $p(x_i) = y_i$ for all $i \in \{0, 1, 2, 3\}$.

   Justification:

2. Assume that $x_0 < x_1 < x_2$ and y_0, y_1, y_2 are given real numbers. Then, there can exist *at most one* polynomial $p \in \mathcal{P}_2$ such that $p(x_i) = y_i$ for all $i \in \{0, 1, 2\}$.

   Justification:

3. Let $p \in \mathcal{P}_d$ be a polynomial of degree $d > 0$, and let $q \colon \mathbf{R} \to \mathbf{R}$ be given by $q(x) = p(x+1) - p(x)$. Then it holds that $q \in \mathcal{P}_{d-1}$.

   Justification:

4. For $n \in \mathbf{N}$, let $x_i^n = i/n$ for $i = 0, 1, \dots, n$. Assume that $u \colon \mathbf{R} \to \mathbf{R}$ is the smooth function given by $u(x) = \sin(3x) + x^3$, and let $p_n \in \mathcal{P}_n$ denote the polynomial interpolation of u at the points $x_0^n, x_1^n, \dots, x_n^n$. Then it holds that

   $$\lim_{n \to \infty} \left(\max_{x \in [0,1]} \left| u(x) - p_n(x) \right| \right) = 0\,.$$

   Justification:

⚙ **Question 3** (Interpolation, open-ended question, **5 marks**). In polynomial interpolation, the error depends both on the function being interpolated and the choice of interpolation nodes. Consider two families of nodes on the interval $[-1, 1]$:

- Equally spaced nodes,
- Chebyshev nodes.

Discuss qualitatively (and illustrate with examples or plots if you wish) how the interpolation error behaves as the degree n increases in each case. Why does one choice of nodes perform better for large n, and what is the mathematical motivation for using Chebyshev nodes? *You may refer to Runge's phenomenon, but go beyond merely stating it.*

Implementation exercise 1 (Interpolation, **5 marks**). Write Julia code that computes and plots the interpolating polynomial $p \in \mathcal{P}_3$ through the following points: $(0, 0)$, $(1, 4)$, $(2, 15)$, $(3, 40)$. The plot should display both the interpolation points and the graph of the interpolating polynomial over an appropriate range. Do not use any other library than the ones already imported.

```julia
using LinearAlgebra
using Plots
# Write your code here
```

Question 4 (Numerical integration, **5 marks**). Are the following statements true or false? Justify briefly.

1. The degree of precision of the following quadrature rule is 2:

$$\int_{-1}^{1} u(x)\,\mathrm{d}x \approx 2u(0)\,.$$

Justification:

2. The degree of precision of the following rule is equal to 3:

$$\int_{-1}^{1} u(x)\,\mathrm{d}x \approx u\left(-\frac{1}{3}\right) + u\left(\frac{1}{3}\right).$$

Justification:

3. For any natural number $N > 0$, there exists a quadrature rule with a degree of precision equal to $2N-1$ of the form

$$\int_{-1}^{1} u(x)\,\mathrm{d}x \approx \sum_{n=1}^{N} w_n u(x_n).$$

Justification:

4. Let $x_i^N = i/N$ and consider the following approximation of $\int_0^1 u(x)\,\mathrm{d}x$:

$$\widehat{I}_N = \frac{1}{2N}\Big(u\left(x_0^N\right) + 2u\left(x_1^N\right) + 2u\left(x_2^N\right) + \ldots + 2u\left(x_{N-2}^N\right) + 2u\left(x_{N-1}^N\right) + u\left(x_N^N\right)\Big). \tag{1}$$

Suppose first that u is the Runge function, given by $u(x) = (1+25x^2)^{-1}$. Then $\widehat{I}_N$ diverges in the limit $N \to \infty$.

Justification:

5. Let $u(x) = \cos(3x)$ and let $\widehat{I}_N$ be as in (1). Then it holds that

$$\lim_{N\to+\infty}\left(\left|\widehat{I}_N - \int_0^1 u(x)\,\mathrm{d}x\right|\right) = 0.$$

Justification:

6. (**Bonus.**) Fix $u(x) = 2x - 1$ and let $\widehat{I}_N$ be as in (1). Then $\widehat{I}_N = 0$ for all $N \geqslant 2$.

Justification:

Question 5 (Gaussian–Hermite numerical integration, **10 marks**). The Gauss–Hermite quadrature formula with n nodes is an approximation of the form

$$I(u) := \int_{-\infty}^{\infty} u(x)\,\mathrm{e}^{-x^2}\,\mathrm{d}x \approx \sum_{i=1}^{n} w_i\, u(x_i) =: \widehat{I}_n(u),$$

which is exact when u is a polynomial of degree $\leq 2n-1$. Note that the nodes are numbered $1, \dots, n$. For this question, we take for granted that, for integers $i \geq 0$, it holds that

$$\int_{-\infty}^{\infty} x^i \mathrm{e}^{-x^2}\,\mathrm{d}x = \begin{cases} 0, & \text{if } i \text{ is odd,} \\ (i-1)!!\sqrt{\dfrac{\pi}{2^i}}, & \text{if } i \text{ is even,} \end{cases}$$

where $(i-1)!! := 1 \times 3 \times 5 \times \cdots \times (i-1)$. In particular, with all the integrals being over $(-\infty, \infty)$, the following special cases may be useful in your computations:

$$\int \mathrm{e}^{-x^2}\,\mathrm{d}x = \sqrt{\pi}, \quad \int x^2\,\mathrm{e}^{-x^2}\,\mathrm{d}x = \frac{\sqrt{\pi}}{2}, \quad \int x^4\,\mathrm{e}^{-x^2}\,\mathrm{d}x = \frac{3}{4}\sqrt{\pi}, \quad \int x^6\,\mathrm{e}^{-x^2}\,\mathrm{d}x = \frac{15}{8}\sqrt{\pi}\,.$$

1. (**5 marks**) Find the nodes and weights of the Gauss–Hermite rule with $n = 3$ nodes. By symmetry, we expect nodes of the form $(-z, 0, z)$ and weights (w_1, w_2, w_1), which reduces the number of unknowns to three.

   Your answer:

2. (**5 marks**) Let $\{H_0, H_1, \dots\}$ denote orthogonal polynomials for the inner product

$$\langle f, g \rangle := \int_{-\infty}^{\infty} f(x) g(x) \, \mathrm{e}^{-x^2} \, \mathrm{d}x$$

which, in addition, satisfy the following two conditions:

- For all $i \in \mathbf{N}$, the polynomial H_i is of degree i.
- The leading coefficient of H_i, which multiplies x^i, is equal to 1.

Calculate H_0, H_1, H_2 and H_3. What is the relationship between H_3 and the quadrature rule found in the first item?

Your answer:

3. (Bonus, **2 marks**) Calculate H_4 and, using this result, deduce the nodes and weights of the Gauss–Hermite quadrature with 4 points.

   Your answer:

Implementation exercise 2 (Numerical integration, **10 marks**). The midpoint quadrature rule reads

$$\int_{-1}^{1} u(x) \, \mathrm{d}x \approx 2u(0) \, .$$

- (**3 marks**) Write a function `midpoint(u, a, b)` that returns, using this quadrature rule, an approximation of the integral

$$\int_{a}^{b} u(x) \, \mathrm{d}x \, . \tag{2}$$

```
function midpoint(u, a, b)
    # Write your code here






end
```

- (**4 marks**) Write a function `composite_midpoint(u, a, b, N)` that returns an approximation of the integral (2), this time using a composite version of the midpoint rule. More precisely, the approximation should be obtained by partitioning the integration interval $[a, b]$ into N cells, and applying the midpoint rule within each cell.

```
function composite_midpoint(u, a, b)
    # Write your code here






end
```

- (**3 marks**) Take $u(x) = \cos(x)$, $a = -1$ and $b = 1$. In this setting, plot the evolution of the error for N varying from 1 to 1000.

```
using Plots
# Write your code here
```