ETH Lecture 401-2673-00L Numerical Methods for **CSE**

Mid-Term Examination

Autumn Term 2021

Thursday, Dec 23, 2021, 10:15, HG F 1

Don't panic!

Family Name		**Grade**
First Name		
Department		
Legi Nr.		
Date	Thursday, Dec 23, 2021	

Points:

Prb. No.	1	2	3	Total
max	15	18	15	48
achvd				

(100% = 48 pts. , ≈40% (passed) = 19 pts.)

- Upon entering the exam room take a seat at a desk on which you find an exam paper with a **red** sticky note "**CSE**" on it.
- This is a **closed-book exam**, no aids are allowed.
- Keep only writing paraphernalia and your ETH ID card on the table.
- Turn off mobile phones, tablets, smartwatches, etc. and stow them away in your bag.
- When told to do so, take the exam paper out of the envelope, and fill in the cover sheet first. Do not turn pages yet!
- Make sure that your exam paper is for the course 401-2673-00L Numerical Methods for CSE, see the top of the front page.
- Turn the cover sheet only when instructed to do so.
- You will be given **10 minutes of advance reading time** to familiarize yourself with the topic areas of the problems. When told, drop any pen! Then you may turn the pages and start reading the problems.
- Start writing only when the start of the exam time proper is announced.
- **Write your answers in the appropriate (green) solution boxes on these problem sheets.**
- **Wrong ticks in multiple-choice boxes can lead to points being subtracted.** Hence, mere guessing is really dangerous! If you have no clue, leave all tickboxes empty.
- If you change your mind about an answer to a (multiple-choice) question, write a clear NO next to the old answer, draw fresh solution boxes/tickboxes and fill them.
- **Anything written outside the answer boxes will not be taken into account.**

- Do not write with red/green color or with pencil.
- **Duration: 30 minutes.**
- When the end of the exam is announced, make sure you have written your name on every sheet.
- The exam proctors will collect the filled exam papers. Please remain seated until they have finished.

Special Covid-19 Safety Measures

- Only students in possession of a valid Covid Certificate are allowed to take the term exam.
- Protective masks covering nose and mouth have to be worn all the time.

Throughout the exam use the notations introduced in class, in particular [Lecture → Section 1.1.1]:

- $(\mathbf{A})_{i,j}$ to refer to the entry of the matrix $\mathbf{A} \in \mathbb{K}^{m,n}$ at position (i, j).
- $(\mathbf{A})_{:,i}$ to designate the ith-column of the matrix $\mathbf{A}$,
- $(\mathbf{A})_{i,:}$ to denote the i-th row of the matrix $\mathbf{A}$,
- $(\mathbf{A})_{i:j,k:\ell}$ to single out the sub-matrix $\left[(\mathbf{A})_{r,s}\right]_{\substack{i\leq r\leq j\\ k\leq s\leq \ell}}$ of the matrix $\mathbf{A}$,
- $(\mathbf{x})_k$ to reference the k-th entry of the vector $\mathbf{x}$,
- $\mathbf{e}_j \in \mathbb{R}^n$ to write the j-th Cartesian coordinate vector,
- $\mathbf{I}$ to denote the identity matrix,
- $\mathbf{O}$ to write a zero matrix,
- $\mathcal{P}_n$ for the space of (univariate polynomials of degree $\leq n$),
- and superscript indices in brackets to denote iterates: $\mathbf{x}^{(k)}$, etc.

By default, vectors are regarded as column vectors.

Problem 0-1: Various Aspects of Interpolation by Global Polynomials

[Lecture → Chapter 5] teaches the mathematical foundations and algorithmic realizations of interpolation by means of global polynomials. This problems reviews some of these aspects.

This is a purely theoretical problem connected with [Lecture → Section 5.2]

Throughout this problem we write $\mathcal{P}_d(\mathbb{R})$ for the space of uni-variate polynomials of degree $\leq d$:

$$\mathcal{P}_k := \{t \mapsto \alpha_k t^k + \alpha_{k-1} t^{k-1} + \cdots + \alpha_1 t + \alpha_0 \cdot 1\,,\ \alpha_j \in \mathbb{R}\}\,. \qquad \text{[Lecture → Eq. (5.2.1.1)]}$$

(0-1.a) (6 pts.) What are the dimensions of the following subspaces of $\mathcal{P}_d(\mathbb{R})$, $d \in \mathbb{N}$?

(i) $V_1 := \{p \in \mathcal{P}_d(\mathbb{R}) : \int_{-1}^{1} p(t)\,\mathrm{d}t = 0\}$: $\dim V_1 =$ ______ .

(ii) $V_2 := \{p \in \mathcal{P}_d(\mathbb{R}) : p(1) = p(-1) = 0\}$: $\dim V_2 =$ ______ .

(iii) $V_3 := \{p \in \mathcal{P}_d(\mathbb{R}) : p^{(3)}(t) = 0\,\forall t \in [-1,1]\}$: $\dim V_3 = \begin{cases} \text{\_\_\_\_\_\_} & \text{for } \text{\_\_\_\_\_\_}\,, \\ \text{\_\_\_\_\_\_} & \text{for } \text{\_\_\_\_\_\_}\,. \end{cases}$

Here $p^{(3)}$ stands for the third derivative of p.

(iv) $V_4 := \{p \in \mathcal{P}_d(\mathbb{R}) : p(t) = p(-t)\}$: $\dim V_4 = \begin{cases} \text{\_\_\_\_\_\_} & \text{for } \text{\_\_\_\_\_\_}\,, \\ \text{\_\_\_\_\_\_} & \text{for } \text{\_\_\_\_\_\_}\,. \end{cases}$

SOLUTION of (0-1.a):

In this problem we can sometimes appeal to the heuristics that

$$\dim V_i = \dim \mathcal{P}_d(\mathbb{R}) - \sharp\{\text{linear constraints defining } V_i\}\,.$$

From [Lecture → Thm. 5.2.1.2] we know that $\dim \mathcal{P}_d = d + 1$.

1. V_1 is defined by a single linear constraint, which means that

$$\dim V_1 = d + 1 - 1 = d\,.$$

2. V_2 is defined by fixing two zeros, which amounts to two linearly independent linear constraints.

$$\dim V_2 = \dim \mathcal{P}_d(\mathbb{R}) - 2 = d - 1\,.$$

Remark. In fact we can write

$$V_2 := \left\{t \mapsto (1-t)^2 q(t),\ q \in \mathcal{P}_{d-2}(\mathbb{R})\right\}.$$

3. If $p \in \mathcal{P}_d(\mathbb{R})$, then for the third derivative $p^{(3)} \in \mathcal{P}_{d-3}(\mathbb{R})$. Moreover, if a polynomial vanishes on an interval ("has infinitely many zeros"), then it must be zero everywhere. We conclude that $p^{(3)} \equiv 0$. As a consequence $p \in \mathcal{P}_2$.

$$\dim V_3 = \begin{cases} 2 & \text{for } d = 1\,, \\ 3 & \text{for } d \geq 2\,,. \end{cases}$$

4. The polynomials in V_4 are *even functions*, which means that they are of the form
$$V_4 \subset \mathcal{P}_n^{\text{even}} := \{t \mapsto a_0 + a_2 t^2 + a_4 t^4 + \cdots + a_{2n} t^{2n}\}\ , \quad n \in \mathbb{N}\ .$$
Elements of space $\mathcal{P}_n^{\text{even}}$ are defined by $n+1$ parameters, which means $\dim \mathcal{P}_n^{\text{even}} = n+1$.

Note that $n = \frac{d}{2}$ for even d and $n = \frac{d-1}{2}$ for odd d.
$$\blacktriangleright \qquad \dim V_4 = \begin{cases} \frac{d}{2}+1 & \text{for even } d\ , \\ \frac{d-1}{2}+1 & \text{for odd } d. \end{cases}$$

▲

(0-1.b) ⚁ (3 pts.) Let $\{t_0, t_1, \ldots, t_n\} \subset \mathbb{R}$, $n \in \mathbb{N}$. The following sets of functions provide bases for $\mathcal{P}_d$:
$$\mathfrak{B}_1 := \left\{ t \mapsto \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{t-t_j}{t_i - t_j}\ ,\quad i = 0,\ldots,n \right\}, \qquad \square$$
$$\mathfrak{B}_2 := \left\{ t \mapsto \prod_{j=0}^{i-1} (t-t_j)\ ,\quad i = 0,\ldots,n \right\}, \qquad \square$$
$$\mathfrak{B}_3 := \left\{ t \mapsto t^i\ ,\quad i = 0,\ldots,n \right\}. \qquad \square$$
Note that an empty product is defined to be $\equiv 1$.

Put the right letter in the box indicating the name of the basis:

A $\hat{=}$ monomial basis , **B** $\hat{=}$ Newton basis , **C** $\hat{=}$ Lagrangian basis .

SOLUTION of (0-1.b):

- $\mathfrak{B}_1$ is the Lagrangian basis **C** [Lecture → Eq. (5.2.2.4)], a cardinal basis for polynomial interpolation in $\{t_0, t_1, \ldots, t_n\}$,
- $\mathfrak{B}_2$ is the Newton basis **B**[Lecture → Eq. (5.2.3.23)], used for update-friendly polynomial interpolation
- $\mathfrak{B}_3$ is the monomial basis **A** [Lecture → Section 5.2.1].

▲

(0-1.c) ⚃ (6 pts.) Given data points (t_i, y_i), $i = 0, \ldots, n$, $n \in \mathbb{N}$, we write $p_{k,\ell}$, $0 \le k \le \ell \le n$, for the polynomial $p_{k,\ell} \in \mathcal{P}_{\ell-k}$ interpolating through $(t_i, y_i)_{i=k}^{\ell}$: $p_{k,\ell}(t_i) = y_i$, $i = k, \ldots, \ell$. Supplement the missing parts of the following recursion:
$$p_{k,\ell}(t) = \frac{\left(\square\right) p_{k+1,\ell}(t) + \left(\square\right) p_{k,\ell-1}(t)}{\square - \square}\ , \quad t \in \mathbb{R}\ , \quad 0 \le k < \ell \le n\ .$$

SOLUTION of (0-1.c):

This is the recursion [Lecture → Eq. (5.2.3.9)] underlying the **Aitken-Neville** scheme:

$$p_{k,\ell}(t) = \frac{\left(t - t_k\right) p_{k+1,\ell}(t) + \left(t_\ell - t\right) p_{k,\ell-1}(t)}{t_\ell - t_k} , \quad t \in \mathbb{R} .$$

The correctness of this recursion is shown by induction, verifying the interpolation conditions.

▲

End Problem 0-1 , 15 pts.

Problem 0-2: Local Representations of Splines

Locally on knot intervals splines coincide with polynomials of a fixed degree. In this problem we compute those local polynomial representations in special cases.

This is a purely theoretical problem based on [Lecture → Section 5.4.1] and [Lecture → Section 5.4.2].

(0-2.a) (6 pts.)

Let s denote a *quadratic spline*, $s \in \mathcal{S}_{2,\mathcal{M}}$ with respect to the knot set $\mathcal{M} := \{-3,-2,-1,1,2,3\}$ and satisfying

$$s \equiv 0 \quad \text{on} \quad [-3,-2] \cup [2,3] \, ,$$
$$s(-1) = s(1) = 1 \, .$$

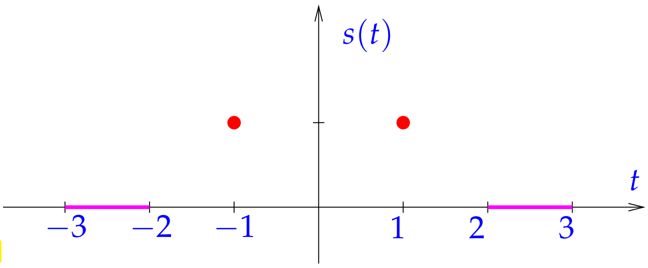

Fig. 1

Determine the unknown real coefficients in the local representations

$$s(t) = \square \, t^2 + \square \, t + \square \qquad \text{for} \quad t \in [1,2] \, ,$$

$$s(t) = \square \, t^2 + \square \, t + \square \qquad \text{for} \quad t \in [-1,1] \, .$$

SOLUTION of (0-2.a):

Since $s \in C^1([-3,3])$ and $s \equiv 0$ on $[2,3]$, we conclude $s(2) = s'(2) = 0$, which means

$$s(t) = \alpha(t-2)^2 \, , \quad \alpha \in \mathbb{R} \quad \text{on} \quad [1,2] \, .$$

From the condition $s(1) = 1$ we get $\alpha = 1$, which implies

$$s(t) = 1 \cdot t^2 - 4 \, t + 4 \quad \text{on} \quad [1,2] \, .$$

As consequence $s'(1) = -2$. Thus the quadratic polynomial p describing s on $[-1,1]$ has to satisfy

$$p(-1) = 1 \quad , \quad p(1) = 1 \quad , \quad p'(1) = -2 \, .$$

This uniquely defines the parabola

$$p(t) = 2 - t^2 \, , \quad t \in \mathbb{R} \, ,$$

from which we infer

$$s(t) = -1 \cdot t^2 + 0 \, t + 2 \quad \text{on} \quad [-1,1] \, .$$

Remark. Here the solution of the problem can stop, but we have not yet verified that the partial s as found so far can be extended to a spline in $\mathcal{S}_{2,\mathcal{M}}$. We conjecture that s is an even function $s(t) = s(-t)$ and, thus, set

$$s(t) = t^2 + 4t + 4 \quad \text{on} \quad [-2,-1] \, .$$

Symmetry arguments readily demonstrate that this completes the definition of the spline s.

▲

(0-2.b) ⚃ (12 pts.) Let the knot set $\mathcal{M} := \{t_0 < t_1 < t_2 < \cdots < t_n\} \subset \mathbb{R}$ be given. The local representation of a cubic spline $s \in \mathcal{S}_{3,\mathcal{M}}$ is given by ($h_j := t_j - t_{j-1}$, $j \in \{1,\dots,n\}$)

$$s|_{[t_{j-1},t_j]}(t) = \alpha_j H_1(\tau) + \beta_j H_2(\tau) + h_j\gamma_j H_3(\tau) + h_j\delta_j H_4(\tau)\,, \quad \tau := \frac{t - t_{j-1}}{h_j}\,, \quad \alpha_j, \beta_j, \gamma_j, \delta_j \in \mathbb{R}\,,$$

where H_1, H_2, H_3, H_4 are the cardinal basis functions for Hermite interpolation on $[0,1]$,

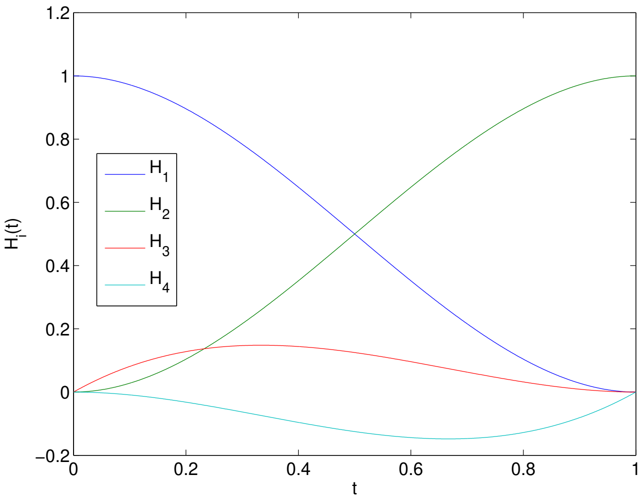

Fig. 2

$$H_1(\tau) = 1 - 3\tau^2 + 2\tau^3\,,$$
$$H_2(\tau) = 3\tau^2 - 2\tau^3\,,$$
$$H_3(\tau) = \tau - 2\tau^2 + \tau^3\,,$$
$$H_4(\tau) = -\tau^2 + \tau^3\,,$$

which satisfy

$$\begin{aligned}
&H_1(0) = 1\,, && H_1(1) = 0\,, && H_1'(0) = 0\,, && H_1'(1) = 0\,, && H_1(\tfrac{1}{2}) = \tfrac{1}{2}\,, && H_1'(\tfrac{1}{2}) = -\tfrac{3}{2}\,,\\
&H_2(0) = 0\,, && H_2(1) = 1\,, && H_2'(0) = 0\,, && H_2'(1) = 0\,, && H_2(\tfrac{1}{2}) = \tfrac{1}{2}\,, && H_2'(\tfrac{1}{2}) = \tfrac{3}{2}\,,\\
&H_3(0) = 0\,, && H_3(1) = 0\,, && H_3'(0) = 1\,, && H_3'(1) = 0\,, && H_3(\tfrac{1}{2}) = \tfrac{1}{8}\,, && H_3'(\tfrac{1}{2}) = -\tfrac{1}{4}\,,\\
&H_4(0) = 0\,, && H_4(1) = 0\,, && H_4'(0) = 0\,, && H_4'(1) = 1\,, && H_4(\tfrac{1}{2}) = -\tfrac{1}{8}\,, && H_4'(\tfrac{1}{2}) = -\tfrac{1}{4}\,.
\end{aligned} \tag{0.2.1}$$

Inserting the midpoints of knot intervals gives us the extended knot set

$$\widetilde{\mathcal{M}} := \{\widetilde{t}_0 < \widetilde{t}_1 < \widetilde{t}_2 < \cdots < \widetilde{t}_{2n}\}\,, \quad \begin{array}{ll} \widetilde{t}_{2j} := t_j\,, & j \in \{0,\dots,n\}\,,\\ \widetilde{t}_{2j-1} := \frac{1}{2}(t_j + t_{j-1})\,, & j \in \{1,\dots,n\}\,. \end{array}$$

On the knot intervals of $\widetilde{\mathcal{M}}$ the spline s has the local representation ($\widetilde{h}_\ell := \widetilde{t}_\ell - \widetilde{t}_{\ell-1}$, $\ell \in \{1,\dots,2n\}$)

$$s|_{[\widetilde{t}_{\ell-1},\widetilde{t}_\ell]}(t) = \widetilde{\alpha}_\ell H_1(\tau) + \widetilde{\beta}_\ell H_2(\tau) + \widetilde{h}_\ell\widetilde{\gamma}_\ell H_3(\tau) + \widetilde{h}_\ell\widetilde{\delta}_\ell H_4(\tau)\,,\ \tau := \frac{t - \widetilde{t}_{\ell-1}}{\widetilde{h}_\ell}\,,\ \widetilde{\alpha}_\ell, \widetilde{\beta}_\ell, \widetilde{\gamma}_\ell, \widetilde{\delta}_\ell \in \mathbb{R}\,.$$

Express the coefficients $\widetilde{\alpha}_\ell, \widetilde{\beta}_\ell, \widetilde{\gamma}_\ell, \widetilde{\delta}_\ell$ in terms of $\alpha_j, \beta_j, \gamma_j, \delta_j$:

$$\widetilde{\alpha}_\ell = \begin{cases} \boxed{\qquad\qquad\qquad\qquad} & \text{for even } \ell = 2j\ , \\ \boxed{\qquad\qquad\qquad\qquad} & \text{for odd } \ell = 2j-1\ , \end{cases}$$

$$\widetilde{\beta}_\ell = \begin{cases} \boxed{\qquad\qquad\qquad\qquad} & \text{for even } \ell = 2j, \\ \boxed{\qquad\qquad\qquad\qquad} & \text{for odd } \ell = 2j-1\ , \end{cases}$$

$$\widetilde{\gamma}_\ell = \begin{cases} \boxed{\qquad\qquad\qquad\qquad} & \text{for even } \ell = 2j, \\ \boxed{\qquad\qquad\qquad\qquad} & \text{for odd } \ell = 2j-1\ , \end{cases}$$

$$\widetilde{\delta}_\ell = \begin{cases} \boxed{\qquad\qquad\qquad\qquad} & \text{for even } \ell = 2j, \\ \boxed{\qquad\qquad\qquad\qquad} & \text{for odd } \ell = 2j-1\ . \end{cases}$$

SOLUTION of (0-2.b):

The key insight is that from (0.2.1) and the chain rule we conclude for the spline s

$$s(t_{j-1}) = \alpha_j\ , \quad s(t_j) = \beta_j\ , \quad s'(t_{j-1}) = \gamma_j\ , \quad s'(t_j) = \delta_j\ , \quad j = 1, \dots n\ . \tag{0.2.2}$$

This means that the coefficients have a concrete meaning as the point values and derivative values of the spline in the endpoints of the knot intervals.

The same argument yields, now using the local representation on the knot intervals of $\widetilde{\mathcal{M}}$:

$$s(\widetilde{t}_{\ell-1}) = \widetilde{\alpha}_\ell\ , \quad s(\widetilde{t}_\ell) = \widetilde{\beta}_\ell\ , \quad s'(\widetilde{t}_{\ell-1}) = \widetilde{\gamma}_\ell\ , \quad s'(\widetilde{t}_\ell) = \widetilde{\delta}_\ell\ , \quad \ell = 1, \dots 2n\ . \tag{0.2.3}$$

Next, we have to compute the values of s and s' at midpoints of knot intervals of $\mathcal{M}$. We start with

$$\begin{aligned} H_1(1/2) &= \tfrac{1}{2}\ , & H_1'(1/2) &= -\tfrac{3}{2}\ ,\ , \\ H_2(1/2) &= \tfrac{1}{2}\ , & H_2'(1/2) &= \tfrac{3}{2}\ ,\ , \\ H_3(1/2) &= \tfrac{1}{8}\ , & H_3'(1/2) &= -\tfrac{1}{4}\ ,\ , \\ H_4(1/2) &= -\tfrac{1}{8}\ , & H_4'(1/2) &= -\tfrac{1}{4}\ ,\ , \end{aligned} \tag{0.2.4}$$

which implies

$$s(\tfrac{1}{2}(t_j + t_{j-1})) = \tfrac{1}{2}(\alpha_j + \beta_j) + \tfrac{1}{8}(\gamma_j - \delta_j)\ , \tag{0.2.5}$$

$$s'(\tfrac{1}{2}(t_j + t_{j-1})) = \tfrac{3}{2}(-\alpha_j + \beta_j) - \tfrac{1}{4}(\gamma_j + \delta_j)\ . \tag{0.2.6}$$

I. For **even** $\ell = 2j$ we have $\widetilde{t}_\ell = t_j$, $j = 1, \dots, n$ and the associated knot interval for $\ell, j > 0$ is

$$[\widetilde{t}_{\ell-1}, \widetilde{t}_\ell] = [\tfrac{1}{2}(t_j + t_{j-1}), t_j]\ .$$

So its right endpoint is a knot of $\mathcal{M}$, the left endpoint a midpoint of a knot interval of $\mathcal{M}$. We conclude from (0.2.2), (0.2.3), (0.2.5)

$$\begin{aligned}
\widetilde{\alpha}_\ell &= s(\widetilde{t}_{\ell-1}) = s(\tfrac{1}{2}(t_j + t_{j-1})) = \tfrac{1}{2}(\alpha_j + \beta_j) + \tfrac{1}{8}(\gamma_j - \delta_j)\ , \\
\widetilde{\beta}_\ell &= s(\widetilde{t}_\ell) = s(t_j) = \beta_j\ , \\
\widetilde{\gamma}_\ell &= s'(\widetilde{t}_{\ell-1}) = s'(\tfrac{1}{2}(t_j + t_{j-1})) = \tfrac{3}{2}(-\alpha_j + \beta_j) - \tfrac{1}{4}(\gamma_j + \delta_j)\ , \\
\widetilde{\delta}_\ell &= s'(\widetilde{t}_\ell) = s'(t_j) = \delta_j\ .
\end{aligned}$$

II. For **odd** $\ell = 2j-1$, $j = 1, \dots, n$, we consider the $\widetilde{\mathcal{M}}$ knot interval

$$[\widetilde{t}_{\ell-1}, \widetilde{t}_\ell] = [t_{j-1}, \tfrac{1}{2}(t_j + t_{j-1})]\ .$$

Now the left endpoint is a midpoint of a knot interval of $\mathcal{M}$, and the right endpoint is a knot of $\mathcal{M}$. As before we conclude

$$\begin{aligned}
\widetilde{\alpha}_\ell &= s(\widetilde{t}_{\ell-1}) = s(t_{j-1}) = \alpha_j\ , \\
\widetilde{\beta}_\ell &= s(\widetilde{t}_\ell) = s(\tfrac{1}{2}(t_j + t_{j-1})) = \tfrac{1}{2}(\alpha_j + \beta_j) + \tfrac{1}{8}(\gamma_j - \delta_j)\ , \\
\widetilde{\gamma}_\ell &= s'(\widetilde{t}_{\ell-1}) = s'(t_{j-1}) = \gamma_j\ , \\
\widetilde{\delta}_\ell &= s'(\widetilde{t}_\ell) = s'(\tfrac{1}{2}(t_j + t_{j-1})) = \tfrac{3}{2}(-\alpha_j + \beta_j) - \tfrac{1}{4}(\gamma_j + \delta_j)\ .
\end{aligned}$$

Summing up

$$\widetilde{\alpha}_\ell = \begin{cases} \tfrac{1}{2}(\alpha_j + \beta_j) + \tfrac{1}{8}(\gamma_j - \delta_j) & \text{for even } \ell\ , \\ \alpha_j & \text{for odd } \ell\ , \end{cases} \qquad \widetilde{\beta}_\ell = \begin{cases} \beta_j & \text{for even } \ell\ , \\ \tfrac{1}{2}(\alpha_j + \beta_j) + \tfrac{1}{8}(\gamma_j - \delta_j) & \text{for odd } \ell\ , \end{cases}$$

$$\widetilde{\gamma}_\ell = \begin{cases} \tfrac{3}{2}(-\alpha_j + \beta_j) - \tfrac{1}{4}(\gamma_j + \delta_j) & \text{for even } \ell\ , \\ \gamma_j & \text{for odd } \ell\ , \end{cases} \qquad \widetilde{\delta}_\ell = \begin{cases} \delta_j & \text{for even } \ell\ , \\ \tfrac{3}{2}(-\alpha_j + \beta_j) - \tfrac{1}{4}(\gamma_j + \delta_j) & \text{for odd } \ell\ . \end{cases}$$

▲

End Problem 0-2 , 18 pts.

Problem 0-3: Aspects of Numerical Quadrature

A quadrature rule approximates the integral of a (continuous) function by means of a weighted sum of function values at so-called quadrature nodes/points. Numerical analysis studies the order of quadrature rules and the asymptotic convergence of the quadrature error as a function of the number of quadrature points.

This problem is linked with [Lecture → Section 7.2], [Lecture → Section 7.4.1], and [Lecture → Section 7.5] and also requires familiarity with [Lecture → § 7.5.0.18].

(0-3.a) (4 pts.) Complete the definition of the order of a quadrature rule:

Definition *cf.* [Lecture → Def. 7.4.1.1]. Order of a quadrature rule

The **order** of a quadrature rule $Q_n : C^0([a,b]) \to \mathbb{R}$ is defined as

$$\text{order}(Q_n) := \boxed{}\left\{m \in \mathbb{N}_0 \colon \ Q_n(\boxed{}) = \int_a^b \boxed{}\,\mathrm{d}t \quad \forall p \in \boxed{}\right\}.$$

SOLUTION of (0-3.a):

Definition *cf.* [Lecture → Def. 7.4.1.1]. Order of a quadrature rule

The **order** of a quadrature rule $Q_n : C^0([a,b]) \to \mathbb{R}$ is defined as

$$\text{order}(Q_n) := \max\left\{m \in \mathbb{N}_0 \colon \ Q_n(\,p\,) = \int_a^b p(t)\,\mathrm{d}t \quad \forall p \in \mathcal{P}_{m-1}\right\}.$$

▲

(0-3.b) (4 pts.) The C++ function

```
unsigned int checkQuadOrder(
const Eigen::VectorXd &c, const Eigen::VectorXd &w,
const double tol = 1.0E-10);
```

expects that the input vectors `c` and `w` have the same length and contain the weights and nodes of a quadrature formula on $[-1,1]$. It returns the order of that quadrature formula. Supplement the missing parts of the following listing by writing valid C++ code in the boxes.

```
unsigned int checkQuadOrder(
const Eigen::VectorXd &c, const Eigen::VectorXd &w,
const double tol = 1.0E-10) {
  const unsigned long N = c.size();
  assert(N == w.size());
  for (unsigned int d = 0; d < [      ] ; ++d) {
    double s = 0.0;
    for (int j = 0; j < [     ] ; ++j) {
```

```
      s += [____] * std::pow([____] , d);
    }
    double val = (d % 2 == 0) ? [____] : [____] ;
    if (std::abs(s - val) > tol) {
      return [____] ;
    }
  }
  return [____] ;
}
```

HINT 1 for (0-3.b):

- The C++ operator `%` gives the remainder of integer division: `p % q` $\widehat{=}\, p \mod q$.
- `std::pow(x,n)` computes x^n for $n \in \mathbb{N}_0$, when x is a floating-point type, and n a cardinal number. By specification `std::pow(0.0,0)== 1.0`.

Writing $Q(f)$ for the evaluation of the quadrature formula for a function f, we have to find the largest $d \in \mathbb{N}_0$ such that

$$Q(\{t \mapsto t^d\}) = \int_{-1}^{1} t^d \mathrm{d}t = \begin{cases} 2/d+1 & \text{for even } d\,, \\ 0 & \text{for odd } d\,. \end{cases}$$

SOLUTION of (0-3.b):

C++ code 0.3.1: Implementation of function `checkQuadOrder()`

```cpp
unsigned int checkQuadOrder(const Eigen::VectorXd &c, const Eigen::VectorXd &w,
                            const double tol = 1.0E-10) {
  const unsigned long N = c.size(); // Number of quadrature points
  assert(N == w.size());            // Numbers of nodes and weights must
      agree
  // Check for exactness for polynomials of degree d
  // Note that the highest possible order is 2N!
  for (unsigned int d = 0; d < 2 * N; ++d) {
    // j-loop performs the evaluation of the quadrature formula
    // for the monomial t ↦ t^d.
    double s = 0.0;
    for (int j = 0; j < N; ++j) {
      s += w[j] * std::pow(c[j], d);
    }
    // Exact value: 2/(d+1) for even degree, zero else.
    double val = (d % 2 == 0) ? 2.0 / (d + 1) : 0.0;
    // Safe test for equality of computed floating point numbers
    if (std::abs(s - val) > tol) {
      return d;
    }
  }
  // The quadrature rule has the highest possible order
  return 2 * N;
}
```

▲

(0-3.c) (3 pts.) The C++ function

```
template <typename FUNCTOR>
double trapezoidalRule(
FUNCTOR &&f, double a, double b, unsigned int N);
```

evaluates the **equidistant trapezoidal quadrature rule** with `N` nodes on the interval $[a, b]$ for a (continuous) function passed through the functor object `f`.

Write valid C++ code in the boxes of the following listing so that the function meets this specification.

```
template <typename FUNCTOR>
double trapezoidalRule(
FUNCTOR &&f, double a, double b, unsigned int N) {
  assert(a < b);
  assert(N >= 2);
  const double h = [          ] ;
  double t = a+h;
  double s = [          ] ;
  for (int j = 1; j < [      ] ; ++j) {
    s += [          ] ;
    t += h;
  }
  s += [        ] ;
  return [        ] ;
}
```

HINT 1 for (0-3.c):

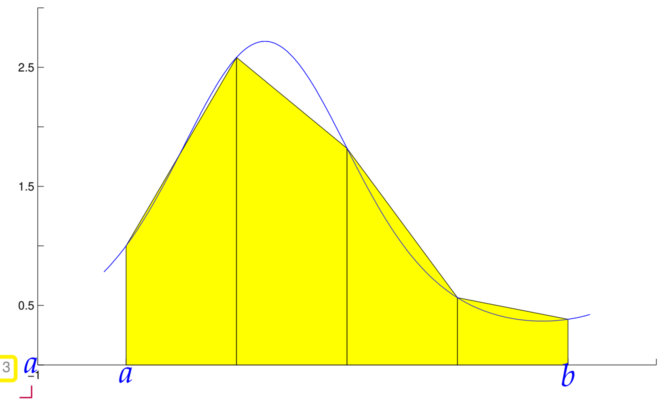

Fig. 3

Recall that the trapezoidal rule approximates the integral $\int_a^b f(t)\,\mathrm{d}t$ by the area under an interpolating polygonal line.

SOLUTION of (0-3.c):

C++ code 0.3.2: Implementation of function `trapezoidalRule()`

```
template <typename FUNCTOR>
double trapezoidalRule(FUNCTOR &&f, double a, double b, unsigned int N) {
  assert(a < b);
  assert(N >= 2);
  // Spacing of the N equidistant quadrature nodes
  const double h = (b - a) / (N - 1);
  // Variable containing location of current quadrature node
  double t = a + h;
  // Left endpoint has weight 0.5*h
  double s = 0.5 * f(a);
  // Summing contributions of N-2 interior quadrature nodes with weight h
  for (int j = 1; j < N - 1; ++j) {
    s += f(t);
    t += h;
  }
  // Right endpoint has weight 0.5*h
  s += 0.5 * f(t);
  // Scaling with h is performed in the end. Can also be done when accumulating
  // value of quadrature formula.
  return h * s;
}
```

▲

(0-3.d) ⚁ (4 pts.) The equidistant trapezoidal quadrature rule as implemented in Sub-problem (0-3.c) is used for the approximate evaluation of the integrals

$$\int_a^b f_{\alpha,\beta}(t)\,\mathrm{d}t\,, \quad a = 0.33,\ b = 1.33\,,$$

where the family of functions $f_{\alpha,\beta} : \mathbb{R} \to \mathbb{R}$ is defined as

$$f_{\alpha,\beta}(t) := \sqrt{|1 + \alpha\sin(\beta t)|}\,, \quad \alpha, \beta, t \in \mathbb{R}\,.$$

The following plots display the modulus of the quadrature error for certain members of this family of functions as a function of the number of quadrature nodes.

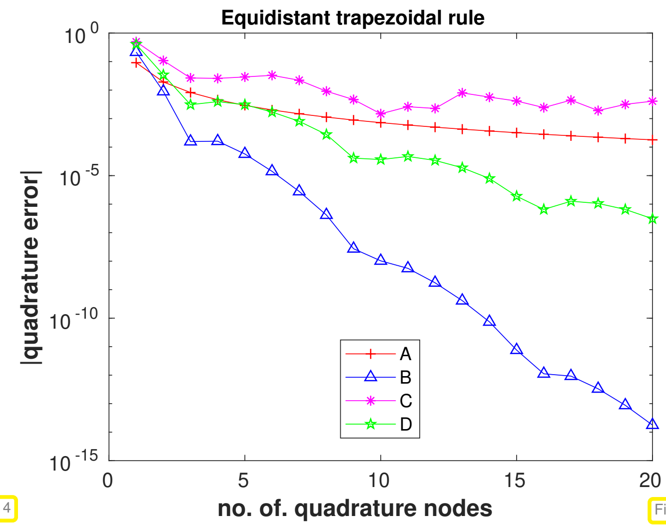

Fig. 4

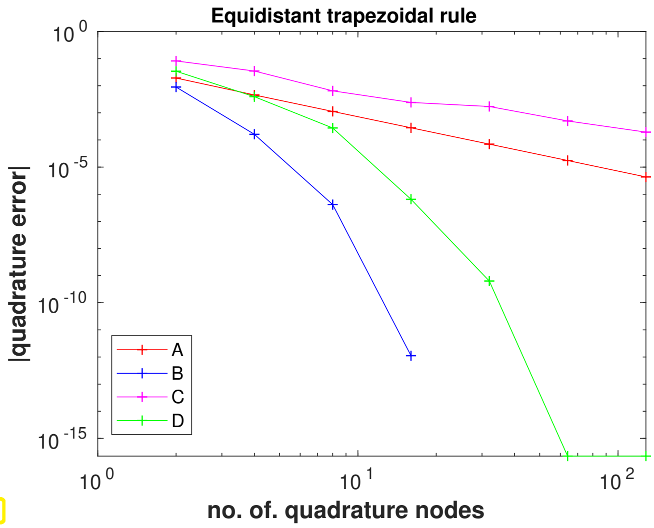

Fig. 5

State, which graph (A,B,C, or D) belongs to which pair of parameters α, β (defining the integrand $f_{\alpha,\beta}$).

Parameters	$\leftrightarrow$	Graph
(1) $\alpha = 0.9, \quad \beta = 2\pi$	$\leftrightarrow$	
(2) $\alpha = 0.5, \quad \beta = \pi$	$\leftrightarrow$	
(3) $\alpha = 1.1, \quad \beta = 2\pi$	$\leftrightarrow$	
(4) $\alpha = 0.5, \quad \beta = 2\pi$	$\leftrightarrow$	

SOLUTION of (0-3.d):

This sub-problem is closely related to [Lecture → Exp. 7.5.0.16] and [Lecture → § 7.5.0.18]. See there for in-depth explanations.

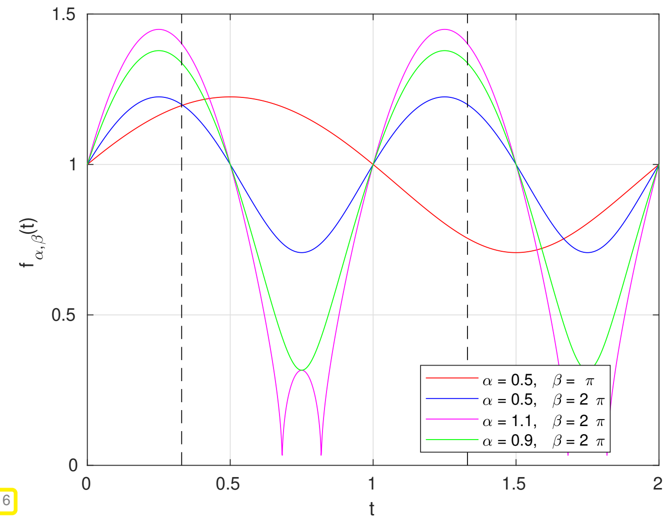

Fig. 6

◁ The four functions from the $f_{\alpha,\beta}$ family, for whom plots of the quadrature errors have been provided.

The colors used correspond to the colors of the associated error graphs.

In the sequel, n stands for the number of quadrature nodes.

- (1) $\alpha = 0.9, \quad \beta = 2\pi$: the function is 1-periodic and analytic in a strip along the real axis. We expect exponential asymptotic convergence for $n \to \infty$, reflected by aligned error points in a linearlogarithmic plot Fig. 4. This expectation is matched by graphs B or D. Since B is already claimed, just D remains.

- (2) $\alpha = 0.5, \quad \beta = \pi$: the function is smooth, but **not** 1-periodic. Thus we can just expect regular $O(n^{-2})$ asymptotic convergence of the trapezoidal rule for $n \to \infty$. By exclusion we conclude that this must correspond to graph A .
- (3) $\alpha = 1.1, \quad \beta = 2\pi$: The function has root-type singularities where $1 + 1.1\sin(2\pi t) = 0$. Hence for this non-smooth integrand we cannot expect full $O(n^{-2})$ algebraic convergence, let alone exponential convergence. This function will result in the slowest observed convergence of the quadrature error, hence graph C .
- (4) $\alpha = 0.5, \quad \beta = 2\pi$: Same as for (1) but with a wider strip of analyticity along the real axis, hence faster exponential convergence than (1), graph B .

The best way to approach this sub-problem is to rank the different functions according to the expected speed of convergence of the quadrature quadrature error. Appealing to the properties of the functions discussed above we obtain:

$$(3) \quad < \quad (2) \quad < \quad (1) \quad < (4)\,,$$

where "$<$" means "expected to converge to zero more slowly than". A corresponding ranking of the error graphs is evident:

$$(C) \quad < \quad (A) \quad < \quad (D) \quad < \quad (B)\,.$$

This yields the right association of functions and graphs.

▲

End Problem 0-3 , 15 pts.