ETH Lecture 401-0663-00L Numerical Methods for PDEs

Mid-term Exam

Spring Term 2023

March 27, 2023, 16:15, HG F1

Don't panic!

Family Name		%
First Name		
Department		
ETH Id Nr.		
Date	March 27, 2023	

Points:

	1	2	3	Total
max	12	10	8	30
achvd				

(100% = 25 pts. , passed = 10 pts.)

General Instructions

- Upon entering the exam room take a seat at a desk on which you find an exam paper with this cover page!
- 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 fill out the cover sheet first. Do not turn pages yet!
- 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. You *must not write* anything during those 10 minutes.
- Start writing only when the start of the exam time proper is announced.
- Do not forget to write your name and ETH ID number on *every* page.
- **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.
- Two blank pages are handed out with the exam: space for personal notes, not graded!
- **Duration: 30 minutes + 10 minutes advance reading time.**
- When the exam is over, drop your pens, tidily arrange your exam paper. Make sure that no page is missing.
- The exam proctors will collect the filled exam papers. Remain seated until they have finished.

Throughout the exam use the notations introduced in class:

- $(\mathbf{A})_{i,j}$ 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 $\mathbf{A}$,
- $\vec{\boldsymbol{\mu}}$, $\vec{\boldsymbol{\varphi}}$, etc., for coefficient vectors,
- $(\vec{\boldsymbol{\mu}})_k$ to reference the k-th entry of the vector $\vec{\boldsymbol{\mu}}$,
- $\mathbf{e}_j \in \mathbb{R}^n$ 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 $\leq n$),
- $\mathcal{S}_p^0(\mathcal{M})$ for degree-p Lagrange finite-element spaces on a mesh $\mathcal{M}$,
- $L^2(\Omega)$, $H^m(\Omega)$, for Sobolev spaces of functions on a domain Ω,
- $\|\cdot\|_X$ for the norm on a function space X,
- and superscript indices in brackets to denote iterates: $\mathbf{x}^{(k)}$, etc.

By default, vectors are regarded as column vectors.

Problem 0-1: From Boundary Value Problems to Variational Problems

[Lecture → Section 1.8] explains how to convert a linear second-order elliptic boundary value problem into an "equivalent" (∗) variational problem. In this problem we practise this conversion.
(∗): "Equivalent" is put in quotation marks because we know from [Lecture → Rem. 1.5.3.10] that, strictly speaking, these two formulations are not equivalent.

This is a purely theoretical problem and it requires familiarity with [Lecture → Section 1.8].
▷ problem name/problem code folder: VPtoBVP

In the following $\Omega \subset \mathbb{R}^2$ designates a two-dimensional bounded computational domain as introduced in [Lecture → § 1.2.1.14]. We also write $\boldsymbol{n} : \partial\Omega \to \mathbb{R}^2$ for its exterior unit normal vector field.

HINT 1 for (0-1.): For all the sub-problems multi-dimensional integration by parts will play a key role. More precisely, you will have to invoke Green's first formula for $d = 2$.

Theorem [Lecture → Thm. 1.5.2.7]. Green's first formula

For all vector fields $\mathbf{j} \in (C^1_{\mathrm{pw}}(\overline{\Omega}))^d$ *and functions* $v \in C^1_{\mathrm{pw}}(\overline{\Omega})$ *holds*

$$\int_\Omega \mathbf{j} \cdot \mathbf{grad}\, v \,\mathrm{d}\boldsymbol{x} = -\int_\Omega \operatorname{div} \mathbf{j}\, v \,\mathrm{d}\boldsymbol{x} + \int_{\partial\Omega} \mathbf{j} \cdot \mathbf{n}\, v \,\mathrm{d}S \,. \qquad \text{[Lecture → Eq. (1.5.2.8)]}$$

(0-1.a) (4 pts.) State, in appropriate Sobolev spaces, the variational problem corresponding to the following second-order elliptic boundary value problem:

$$\frac{\partial}{\partial x_1}\left(e^{x_2}\frac{\partial u}{\partial x_1}(\boldsymbol{x})\right) + \frac{\partial}{\partial x_2}\left(e^{x_1}\frac{\partial u}{\partial x_2}(\boldsymbol{x})\right) = x_1^2 + x_2^2 \quad \text{in} \quad \Omega \quad , \quad u = 0 \quad \text{on} \quad \partial\Omega \;? \qquad (0.1.1)$$

$$u \in \boxed{} : \boxed{} = \boxed{} \quad \forall v \in \boxed{} \,.$$

SOLUTION of (0-1.a):

It is important to notice that the PDE in (0.1.1) can be expressed using the differential operators [Lecture → § 0.3.2.19]

$$\operatorname{div} \mathbf{j}(\boldsymbol{x}) := \frac{\partial j_1}{\partial x_1}(\boldsymbol{x}) + \frac{\partial j_2}{\partial x_2}(\boldsymbol{x}) \quad \text{and} \quad \mathbf{grad}\, v(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial v}{\partial x_1}(\boldsymbol{x}) \\ \frac{\partial v}{\partial x_2}(\boldsymbol{x}) \end{bmatrix} .$$

We get

$$\frac{\partial}{\partial x_1}\left(e^{x_2}\frac{\partial u}{\partial x_1}(\boldsymbol{x})\right) + \frac{\partial}{\partial x_2}\left(e^{x_1}\frac{\partial u}{\partial x_2}(\boldsymbol{x})\right) = \operatorname{div}\left(\begin{bmatrix} e^{x_2} & 0 \\ 0 & e^{x_1} \end{bmatrix} \mathbf{grad}\, u\right)(\boldsymbol{x}) \,.$$

Then we can apply the "algorithm" of [Lecture → Section 1.8] as in [Lecture → Ex. 1.8.0.2]. We test with $v \in C_0^\infty(\Omega)$ and integrate, which yields

$$\int_\Omega \operatorname{div}\left(\begin{bmatrix} e^{x_2} & 0 \\ 0 & e^{x_1} \end{bmatrix} \mathbf{grad}\, u\right)(\boldsymbol{x}) v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = \int_\Omega (x_1^2 + x_2^2) v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \quad \forall v \in C_0^\infty(\Omega) \,.$$

We test with functions vanishing on $\partial\Omega$, because the Dirichlet boundary conditions already fix u there.

Then we integrate by parts by means of [Lecture → Thm. 1.5.2.7] with $\mathbf{j} := \begin{bmatrix} e^{x_2} & 0 \\ 0 & e^{x_1} \end{bmatrix} \mathbf{grad}\, u$ and arrive at

$$-\int_\Omega \begin{bmatrix} e^{x_2} & 0 \\ 0 & e^{x_1} \end{bmatrix} \mathbf{grad}\, u(\boldsymbol{x}) \cdot \mathbf{grad}\, v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = \int_\Omega (x_1^2 + x_2^2) v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \quad \forall v \in C_0^\infty(\Omega) . \tag{0.1.2}$$

Since the matrix function $\boldsymbol{x} \mapsto \begin{bmatrix} e^{x_2} & 0 \\ 0 & e^{x_1} \end{bmatrix}$ is uniformly positive definite, the energy norm induced by the bilinear form of (0.1.2) is equivalent to $|\cdot|_{H^1(\Omega)}$. Moreover, the homogeneous Dirichlet boundary conditions are **essential** [Lecture → Section 1.9] and will be imposed on both trial and test space. Thus $H_0^1(\Omega)$ is the right Sobolev space and the final variational problem reads

$$u \in H_0^1(\Omega): \quad -\int_\Omega \begin{bmatrix} e^{x_2} & 0 \\ 0 & e^{x_1} \end{bmatrix} \mathbf{grad}\, u(\boldsymbol{x}) \cdot \mathbf{grad}\, v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = \int_\Omega (x_1^2 + x_2^2) v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \quad \forall v \in H_0^1(\Omega) .$$

Equivalently, this can be written as

$$u \in H_0^1(\Omega): \quad -\int_\Omega e^{x_2} \frac{\partial u}{\partial x_1}(\boldsymbol{x}) \frac{\partial v}{\partial x_1}(\boldsymbol{x}) + e^{x_1} \frac{\partial u}{\partial x_2}(\boldsymbol{x}) \frac{\partial v}{\partial x_2}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = \int_\Omega \|\boldsymbol{x}\|^2 v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \quad \forall v \in H_0^1(\Omega) .$$

▲

(0-1.b) (4 pts.) Write down the variational problem induced by the second-order elliptic boundary value problem

$$-\Delta u = 1 \quad \text{in} \quad \Omega \quad , \quad -\mathbf{grad}\, u \cdot \boldsymbol{n} = u \quad \text{on} \quad \partial\Omega . \tag{0.1.3}$$

in suitable Sobolev spaces:

$$u \in \boxed{} : \boxed{} = \boxed{} \quad \forall v \in \boxed{} .$$

SOLUTION of (0-1.b):

The boundary value problem (0.1.3) agrees with that considered in [Lecture → Ex. 1.8.0.6] for $f \equiv 1$ and $\Psi(u) = u$.

Since u is not known on $\partial\Omega$ we have to test the PDE of (0.1.3) with $v \in C^\infty(\Omega)$ and get

$$\int_\Omega (-\Delta u)(\boldsymbol{x}) v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = \int_\Omega v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \quad \forall v \in C^\infty(\Omega) .$$

Next we carry out integration by parts according to [Lecture → Thm. 1.5.2.7], which yields

$$\int_\Omega \mathbf{grad}\, u(\boldsymbol{x}) \cdot \mathbf{grad}\, v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} - \int_{\partial\Omega} \mathbf{grad}\, u(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x}) \, v(\boldsymbol{x}) \,\mathrm{d}S(\boldsymbol{x}) = \int_\Omega v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \quad \forall v \in C^\infty(\Omega) .$$

Finally, we recast the boundary term using the boundary conditions and end up with

$$\int_\Omega \mathbf{grad}\, u(\boldsymbol{x}) \cdot \mathbf{grad}\, v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} + \int_{\partial\Omega} u(\boldsymbol{x}) \, v(\boldsymbol{x}) \,\mathrm{d}S(\boldsymbol{x}) = \int_\Omega v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \quad \forall v \in C^\infty(\Omega) .$$

Obviously, this variational problem is naturally posed in the "energy space" $H^1(\Omega)$.

$$u \in H^1(\Omega): \quad \int_\Omega \mathbf{grad}\, u(x) \cdot \mathbf{grad}\, v(x)\, \mathrm{d}x + \int_{\partial\Omega} u(x)\, v(x)\, \mathrm{d}S(x) = \int_\Omega v(x)\, \mathrm{d}x \quad \forall v \in H^1(\Omega) .$$

▲

(0-1.c) (4 pts.) Given a uniformly positive continuous function $\alpha \in C^0(\overline{\Omega})$, convert the second-order *vectorial* boundary value problem

$$-\mathbf{grad}(\alpha(x)\,\mathrm{div}\,\mathbf{u}(x)) + \mathbf{u}(x) = 0 \quad \text{in} \quad \Omega \quad , \quad \alpha(x)\,\mathrm{div}\,\mathbf{u}(x) = 1 \quad \text{on} \quad \partial\Omega \, , \tag{0.1.4}$$

into a linear variational problem posed in a suitable Sobolev space

$$\mathbf{u} \in V: \quad \boxed{} = \boxed{} \quad \forall v \in V \, ,$$

$$V := \boxed{} \, .$$

SOLUTION of (0-1.c):

The PDE in (0.1.4) expresses an equation of vector fields $\Omega \mapsto \mathbb{R}^2$. Therefore, it has to be tested with a vector field $\mathbf{v} \in \boldsymbol{C}^\infty(\Omega) := (C^\infty(\Omega))^2$. Note that $\mathbf{u}$ is not immediately known on $\partial\Omega$ so that we test with functions that do not vanish on (any part of) the boundary $\partial\Omega$. After integrating over Ω we get

$$\int_\Omega (-\mathbf{grad}(\alpha(x)\,\mathrm{div}\,\mathbf{u}(x)) + \mathbf{u}(x))\mathbf{v}(x)\, \mathrm{d}x = 0 \quad \forall \mathbf{v} \in \boldsymbol{C}^\infty(\Omega) \, .$$

Next, we resort to integration by parts using [Lecture → Thm. 1.5.2.7] in "reverse mode" with $\mathbf{j} := \mathbf{v}$ and $v := \alpha(x)\,\mathrm{div}\,\mathbf{u}$:

$$\int_\Omega \mathrm{div}(\alpha(x)\mathbf{u}(x))\ \mathrm{div}\,\mathbf{v}(x) + \mathbf{u}(x)\mathbf{v}(x)\, \mathrm{d}x - \int_{\partial\Omega} \alpha(x)\,\mathrm{div}\,\mathbf{u}(x)(\mathbf{v}(x) \cdot \boldsymbol{n}(x))\, \mathrm{d}x = 0 \quad \forall \mathbf{v} \in \boldsymbol{C}^\infty(\Omega) \, .$$

We use the boundary conditions from (0.1.4) to rephrase the boundary term and move it to the right-hand side.

$$\underbrace{\int_\Omega \mathrm{div}(\alpha(x)\mathbf{u}(x))\ \mathrm{div}\,\mathbf{v}(x) + \mathbf{u}(x)\mathbf{v}(x)\, \mathrm{d}x}_{=:\mathrm{a}(\mathbf{u},\mathbf{v})} = \int_{\partial\Omega} 1\,(\mathbf{v}(x) \cdot \boldsymbol{n}(x))\, \mathrm{d}x \quad \forall \mathbf{v} \in \boldsymbol{C}^\infty(\Omega) \, . \tag{0.1.5}$$

The energy norm spawned by (0.1.5) is

$$\|\mathbf{v}\|_{\mathrm{a}}^2 := \int_\Omega |\,\mathrm{div}\,\mathbf{v}(x)|^2 + \|\mathbf{v}(x)\|^2\, \mathrm{d}x \, .$$

The right Sobolev space is the vector space of vector fields with bounded energy norm

$$V = \left\{ \mathbf{v} : \Omega \to \mathbb{R}^2 : \int_\Omega |\,\mathrm{div}\,\mathbf{v}(x)|^2 + \|\mathbf{v}(x)\|^2\, \mathrm{d}x < \infty \right\} \, .$$

The final variational formulations is just (0.1.5) posed on V:

$$\mathbf{u} \in V\colon \quad \boxed{\int_{\Omega} \operatorname{div}(\alpha(\boldsymbol{x})\mathbf{u}(\boldsymbol{x}))\, \operatorname{div}\mathbf{v}(\boldsymbol{x}) + \mathbf{u}(\boldsymbol{x})\mathbf{v}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_{\partial\Omega} \mathbf{v}(\boldsymbol{x})\cdot\boldsymbol{n}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \quad \forall \mathbf{v} \in V}\ .$$

▲

End Problem 0-1 , 12 pts.

Problem 0-2: Lagrangian Finite Elements on Criss-Cross Meshes

In this problem we study Lagrangian finite element spaces [Lecture → Section 2.6] and related stiffness matrices on a special family of meshes.

Th problem assumes knowledge about the contents of [Lecture → Section 2.6].
▷ problem name/problem code folder: FESpacesCrissCross

The figure beside shows a so-called **criss-cross mesh** $\mathcal{M}_M$ of the unit square $\Omega :=]0,1[^2$, which arises from splitting every square of an $M \times M$, $M \in \mathbb{N}$, $M \geq 3$, tensor-product mesh into four congruent isosceles triangles.

criss-cross mesh $\mathcal{M}_M$ ▷

Marked in magenta in the figure is the part of the boundary

$$\Gamma_D := (\{0\} \times [0,1]) \cup (\{1\} \times [0,1]) .$$

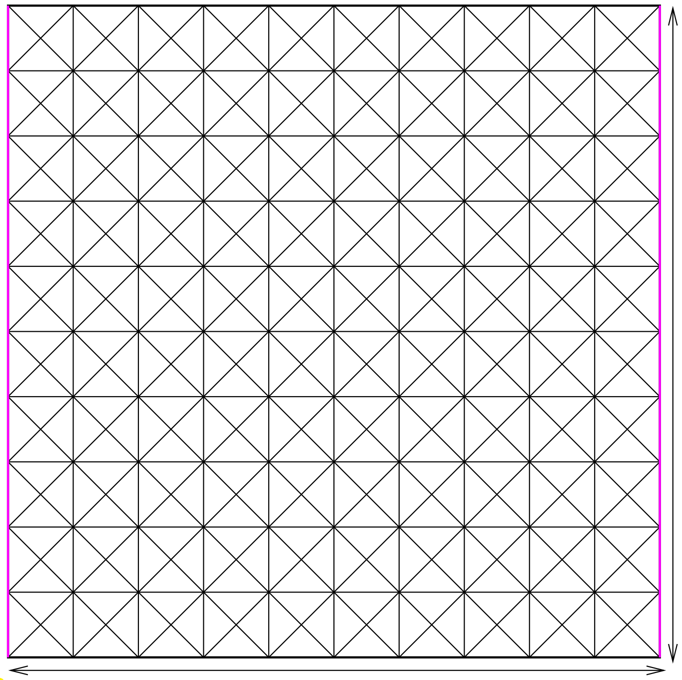


Fig. 1

(0-2.a) (4 pts.) What are the dimensions of the following Lagrange finite-element spaces on the criss-cross mesh $\mathcal{M}_M$ of Fig. 1, $M \in \mathbb{N}$, $M \geq 3$?

$$\dim \mathcal{S}_1^0(\mathcal{M}_M) = \boxed{} , \tag{0.2.1}$$

$$\dim \mathcal{S}_2^0(\mathcal{M}_M) = \boxed{} , \tag{0.2.2}$$

$$\dim V = \boxed{} , \quad V := \{v_h \in \mathcal{S}_1^0(\mathcal{M}_M) : \; v_h|_{\Gamma_D} = 0\} , \tag{0.2.3}$$

$$\dim W = \boxed{} , \quad W := \{v_h \in \mathcal{S}_2^0(\mathcal{M}_M) : \; v_h|_{\Gamma_D} = 0\} . \tag{0.2.4}$$

SOLUTION of (0-2.a):

- As explained in [Lecture → Section 2.4.2] the dimension of $\mathcal{S}_1^0(\mathcal{M})$ on a triangular mesh $\mathcal{M}$ agrees with the number of nodes.

$$\dim \mathcal{S}_1^0(\mathcal{M}_M) = (M+1)^2 + M^2 = 2M^2 + 2M + 1 .$$

- The canonical basis functions of $\mathcal{S}_2^0(\mathcal{M}_M)$ are associated with the nodes and edges of the mesh

[Lecture → Ex. 2.6.1.2], which means

$$\dim \mathcal{S}_2^0(\mathcal{M}_M) = \underbrace{(M+1)^2 + M^2}_{\sharp\text{nodes}} + \underbrace{2M(M+1) + 4M^2}_{\sharp\text{edges}} = 8M^2 + 4M + 1 \;,$$

because there are $2M(M+1)$ edges of the tensor-product mesh plus $4M^2$ edges inside the squares.

- In the case of V we drop all basis functions associated with nodes $\in \Gamma_D$. There are $2(M+1)$ such nodes, which implies

$$\dim V = \dim \mathcal{S}_1^0(\mathcal{M}_M) - 2(M+1) = (M+1)^2 + M^2 - 2(M+1) = 2M^2 - 1 \;.$$

- The space W emerges from $\mathcal{S}_2^0(\mathcal{M}_M)$ by dropping all canonical basis functions belonging to nodes or edges in Γ_D:

$$\dim W = \dim \mathcal{S}_2^0(\mathcal{M}_M) - 2(M+1) - 2M = 8M^2 + 4M + 1 - 4M - 2 = 8M^2 - 1 \;.$$

▲

From now consider the discrete variational problem

$$u \in V\colon \quad \int_\Omega u(\boldsymbol{x})v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_{\partial\Omega} v(\boldsymbol{x})\,\mathrm{d}S(\boldsymbol{x}) \quad \forall v \in V \;, \tag{0.2.5}$$

where V is defined in (0.2.3).

(0-2.b) (4 pts.) Compute the following (maximal/minimal) numbers of non-zero elements of (rows of) the Galerkin matrix $\mathbf{A}$ for (0.2.5) in terms of M, assuming that the customary minimally locally supported "tent" basis functions [Lecture → Section 2.4.3] are used.

$$\max\{\sharp\{j \in \{1,\dots,\dim V\} : (\mathbf{A})_{i,j} \neq 0\},\ i \in \{1,\dots,\dim V\}\} = \boxed{} \;,$$

$$\min\{\sharp\{j \in \{1,\dots,\dim V\} : (\mathbf{A})_{i,j} \neq 0\},\ i \in \{1,\dots,\dim V\}\} = \boxed{} \;,$$

$$\operatorname{nnz}(\mathbf{A}) := \sharp\{(i,j) \in \{1,\dots,\dim V\}^2 : (\mathbf{A})_{i,j} \neq 0\} = \boxed{} \;.$$

SOLUTION of (0-2.b):

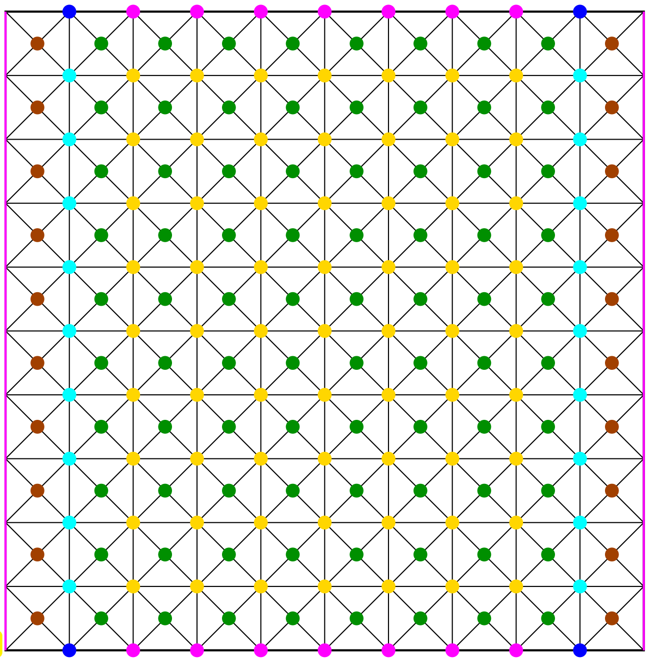
Rows of the Galerkin matrix are associated with nodes of the mesh that do not lie on Γ_D. The number of non-zero entries in row i is the number of edges connecting node i with other nodes $\notin \Gamma_D$.

The figure shows the nodes $\notin \Gamma_0$ colored according to the number of "active" neighboring nodes:

- yellow nodes: eight neighbors,
- cyan nodes: seven neighbors,
- magenta nodes: five neighbors,
- green nodes: four neighbors,
- brown nodes: two neighbors.

Every row of $\mathbf{A}$ belonging to a an active node with k active neighbors has exactly k non-zero entries

Fig. 2



- The maximal number of non-zero entries in a row of $\mathbf{A}$ is attained for rows belonging to yellow nodes, which have eight active neighbors

$$\max\{\sharp\{j \in \{1,\dots,\dim V\} : (\mathbf{A})_{i,j} \neq 0\},\ i \in \{1,\dots,\dim V\}\} = \boxed{9}\ .$$

- Brown nodes with only two active neighbors supply rows with a minimal number of non-zero entries

$$\min\{\sharp\{j \in \{1,\dots,\dim V\} : (\mathbf{A})_{i,j} \neq 0\},\ i \in \{1,\dots,\dim V\}\} = \boxed{3}\ .$$

- The number of non-zero entries of the Galerkin matrix can be read off Fig. 2 as number of active nodes plus the total number of their active neighbors. To that end we have to count the nodes of different color:
  - $\sharp\{$yellow nodes$\} = (M-1)(M-3)$ (eight neighbors),
  - $\sharp\{$cyan nodes$\} = 2(M-1)$ (seven neighbors),
  - $\sharp\{$magenta nodes$\} = 2(M-3)$ (five neighbors),
  - $\sharp\{$green nodes$\} = M(M-2)$ (four neighbors),
  - $\sharp\{$blue nodes$\} = 4$ (four neighbors),
  - $\sharp\{$brown nodes$\} = 2M$ (two neighbors)

$$\begin{aligned}\boxed{\mathrm{nnz}(\mathbf{A})} &:= \sharp\{(i,j) \in \{1,\dots,\dim V\}^2 :\ (\mathbf{A})_{i,j} \neq 0\} \\ &= \dim V + 8\cdot(M-1)(M-3) + 7\cdot 2(M-1) + 5\cdot 2(M-3) \\ &\quad + 4\cdot M(M-2) + 4\cdot 4 + 2\cdot 2M \\ &\boxed{= 14M^2 - 12M - 5}\ .\end{aligned}$$

Alternative reasoning: If there were no essential boundary conditions, that is on the full space $\mathcal{S}_1^0(\mathcal{M}_M)$ the number of non-zero entries of the "full" Galerkin matrix would be the twice the number $6M^2+2M$ of edges of $\mathcal{M}_M$ (off-diagonal entries) plus the number $2M^2+2M+1$ of nodes of $\mathcal{M}_M$ (diagonal entries), which yields

$$\mathrm{nnz}(\text{full Galerkin matrix}) = 14M^2+6M+1\ .$$

However, nodes on Γ_0 and edges connecting to those must be omitted.

There are

- $2(M+1)$ nodes on Γ_0,
- $2M$ edges on Γ_0,
- $2(M+1)$ horizontal edges emanating from nodes on Γ_0,
- $4M$ skew edges with an endpoint on Γ_0.

In the figure the edges to be dropped are marked with •. Their total number is

$$2M+2(M+1)+4M = 8M+2\ .$$

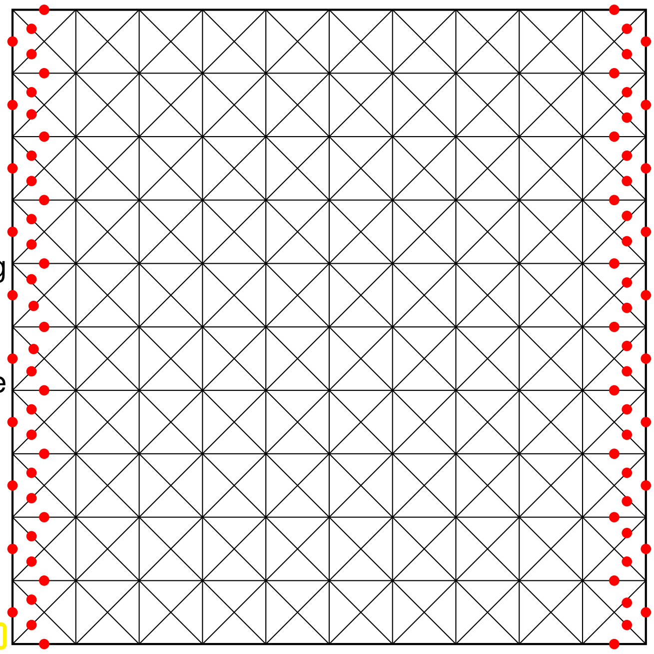


Fig. 3

Summing up, we calculate

$$\boxed{\mathrm{nnz}(\mathbf{A})} = 2\Big(6M^2+2M-(8M+2)\Big)+\Big(2M^2+2M+1-(2M+2)\Big)$$
$$\boxed{=14M^2-12M-5}\ .$$

(0-2.c) (2 pts.) What is the number of non-zero entries of the right-hand-side vector $\vec{\varphi}$ of the linear system of equations arising from (0.2.5) when using the customary minimally locally supported "tent" basis functions of V [Lecture → Section 2.4.3]?

$$\mathrm{nnz}(\vec{\varphi}) = \sharp\{i\in\{1,\dots,\dim V\}:\ (\vec{\varphi})_i\neq 0\} = \boxed{}\ .$$

SOLUTION of (0-2.c):

The entries of $\vec{\varphi}$ are given by the formula

$$(\vec{\varphi})_i = \int_{\partial\Omega} b_h^i\,\mathrm{d}S(\boldsymbol{x})\ ,$$

where $b_h^i\in V$ is a tent basis function associated with a node of $\mathcal{M}_M$ not located on Γ_D. Those nodes are marked with colored dots in Fig. 2.

Hence, $(\vec{\varphi})_i\neq 0$, *if and only if* $\partial\Omega$ cuts through the interior of the support of b_h^i. This is the case only for the tent basis functions sitting on the blue and magenta nodes in Fig. 2, of which there are $2(M-1)$.

$$\sharp\{i\in\{1,\dots,\dim V\}:\ (\vec{\varphi})_i\neq 0\} = \boxed{2(M-1)}\ .$$

This reasoning is valid, because the tent basis functions are strictly positive in the interior of their support.

▲

End Problem 0-2 , 10 pts.

Problem 0-3: `DofHandler` and Assembly

A key component in the cell-oriented assembly of finite-element Galerkin matrices is a local→global index mapping/d.o.f. handler, in LehrFEM++ encoded in **lf::assemble::DofHandler** objects. In this problem we consider such an object for a concrete small mesh and try to figure out what the member functions will return.

This problem is linked to [Lecture → Section 2.7.4.2], in particular [Lecture → § 2.7.4.13]. It requires knowledge about LehrFEM++.

▷ problem name/problem code folder: DOFHandlerAssembly

Assume that in a LehrFEM++-based C++ code the variable `mesh_p` holds a pointer to an object of type **lf::mesh::hybrid2d::Mesh**, which describes the following *2D triangular mesh* with 15 cells, 27 edges, and 13 nodes for the square domain $\Omega :=]0,3[^2$.

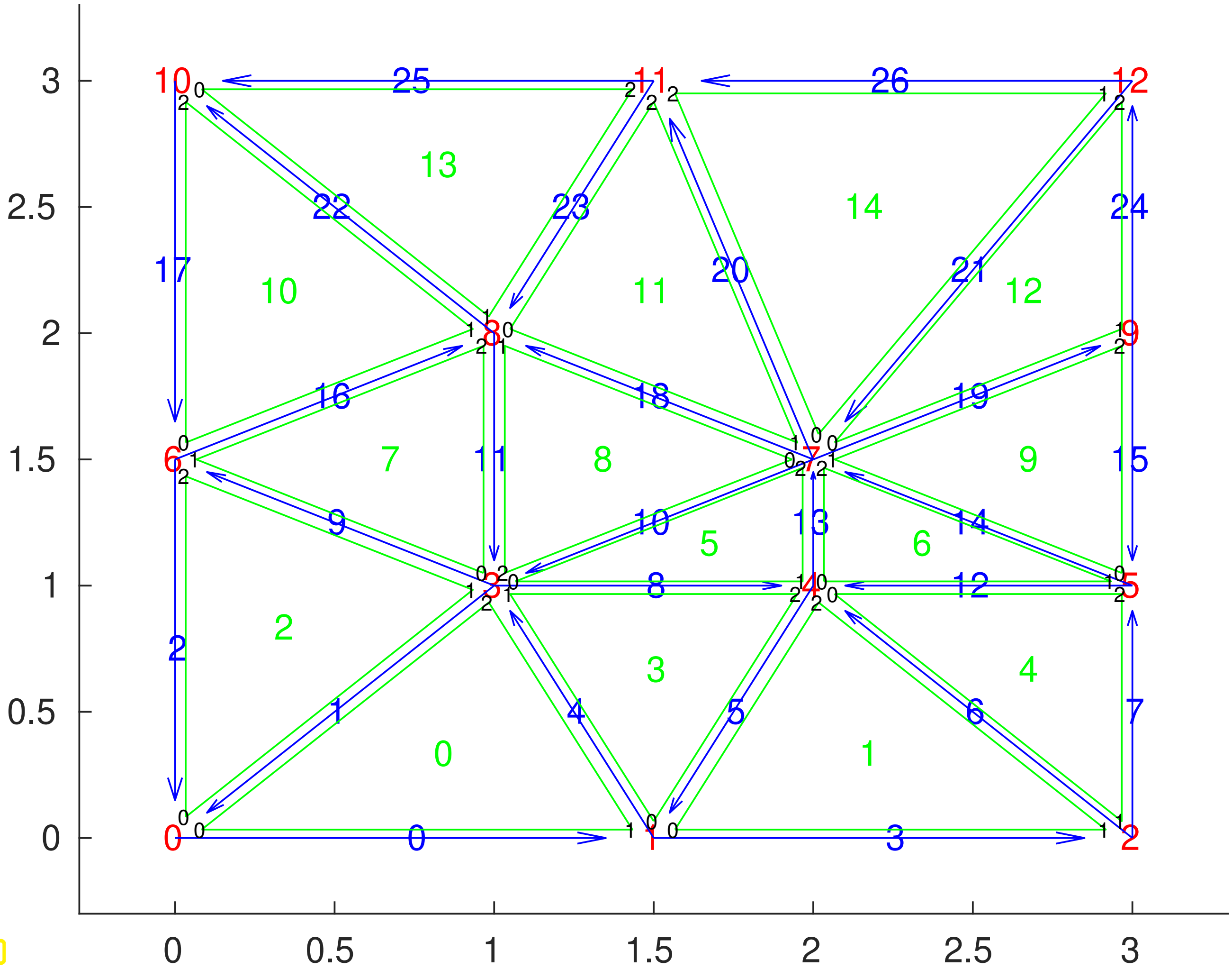


Fig. 4

The big red numbers $\in \{0,\dots,12\}$ give the global index numbers of the mesh nodes as returned by the `Index()` member function of **lf::mesh::Mesh**. The blue numbers $\in \{0,\dots,26\}$ on top of mesh edges give their respective global index numbers, while the green numbers in the center of triangles tell the global index numbers of the mesh cells. The small black numbers at the vertices of every triangle provide their local indices within that triangle.

In the code we initialize the d.o.f. handler object `dofh` as follows:

```
lf::assemble::UniformFEDofHandler dofh(
mesh_p, {{lf::base::RefEl::kPoint(), 0},
  {lf::base::RefEl::kSegment(), 1},
```

```
{lf::base::RefEl::kTria(), 2},
{lf::base::RefEl::kQuad(), 2}});
```

Moreover, below the variable `cell_p`, whenever it occurs, holds a pointer to an object of type **lf::mesh::hybrid2d::Triangle**, which means that

```
cell_p.RefEl() == lf::base::RefEl::kTria()
```

evaluates to **true**.

(0-3.a) (3 pts.) What integer numbers will the following function calls return?

- `dofh.NumDofs()` = ☐ ,
- `dofh.NumLocalDofs(*cell_p)` = ☐ ,
- `dofh.NumInteriorDofs(*cell_p)` = ☐ .

SOLUTION of (0-3.a):

The initialization of the d.o.f. handler object tells us that exactly one global shape function will be associated with every edge of the mesh, while two basis functions will be associated with every cell and none with the nodes. So the total number of global shape function/degrees of freedom managed by `dofh` is the number of edges of the mesh plus twice the number of cells

$$\texttt{dofh.NumDofs()} \;=\; 27 + 2 \cdot 15 = 57\;.$$

A triangle has three edges and, thus, is inside the support of three edge-associated shape functions. Of course it also lies inside the support of its two cell-associated basis functions. So the number of *covering* local shape functions is $3 + 2 = 5$ for every triangle:

$$\texttt{dofh.NumLocalDofs(*cell\_p)} \;=\; 5\;.$$

There are two shape functions *associated* with a triangle:

$$\texttt{dofh.NumInteriorDofs(*cell\_p)} \;=\; 2\;.$$

For details see [Lecture → § 2.7.4.13].

▲

(0-3.b) (5 pts.) Now assume that `cell_p` points to the triangle object for *cell no. 8* of the given mesh of Fig. 4, that is `mesh_p->Index(*cell_p)` returns 8. What sequences of integers $\in \mathbb{N}_0$ are returned by the following function calls? Write them as comma-separated numbers.

- `dofh.GlobalDofIndices(*cell_p)` = (☐)
- `dofh.InteriorGlobalDofIndices(*cell_p)` = (☐) .

HINT 1 for (0-3.b):

LEHRFEM++'s local numbering convention for the edges of a triangular cell is displayed in the figure beside, *cf.* [Lecture → § 2.7.2.14].

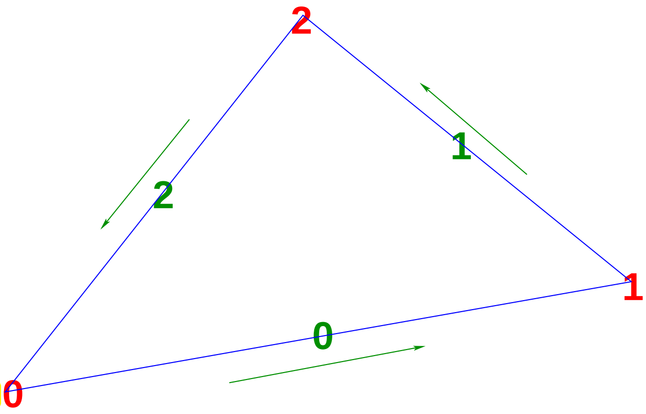


Fig. 5

┘

HINT 2 for (0-3.b): Recall the numbering conventions of LEHRFEM++'s **lf::assemble::DofHandler** from [Lecture → Rem. 2.7.4.17]:

(I) D.o.f. associated with lower-dimensional entities are numbered first:

$$\textbf{POINT} \longrightarrow \textbf{SEGMENT} \longrightarrow \{\textbf{TRIA,QUAD}\}\ .$$

(II) The indices of d.o.f. belonging to entities of the same co-dimension increase with increasing entity indices as returned by the `Index()` function.

These rules govern both the *local and global* numbering of basis functions (local and global shape functions). ┘

SOLUTION of (0-3.b):

The sub-entities of triangle no. 8 have the following indices

$$\begin{aligned} \text{vertices} &\leftrightarrow (7,8,3)\ ,\\ \text{edges} &\leftrightarrow (18,11,10)\ , \end{aligned}$$

where the ordering reflects their local numbering.

According to [Lecture → Rem. 2.7.4.17] reproduced as hint, the basis functions associated with edges are numbered first and that numbering follows their global indices. This means that the indices of the edge-associated basis functions covering triangle 8 are $18, 11, 10$. The cell-associated basis functions are numbered after the edge-associated starting with the index 27 (= total number of edges). The two basis functions for a triangle are numbered consecutively. This means that the cell-associated basis functions for triangle 8 have the indices $27 + 16, 27 + 17$.

► `dofh.GlobalDofIndices(*cell_p) =` $\left(18,11,10,43,44\right)$.

The call to `dofh.InteriorGlobalDofIndices()` just returns the global index numbers of the basis functions associated with the triangle 8:

► `dofh.InteriorGlobalDofIndices(*cell_p) =` $\left(43,44\right)$.

C++ code 0.3.1: Test code

```cpp
void dofProbe() {
  std::cout << "Probing DofHandler output for one cell of test mesh #3"
  << std::endl;
  // Generate test mesh number 3
  constexpr int selector = 3;
  std::shared_ptr<const lf::mesh::Mesh> mesh_p =
  lf::mesh::test_utils::GenerateHybrid2DTestMesh(selector);
  // Create a dof handler object describing a uniform distribution
  // of shape functions
  lf::assemble::UniformFEDofHandler dofh(mesh_p,
  {{lf::base::RefEl::kPoint(), 0},
    {lf::base::RefEl::kSegment(), 1},
    {lf::base::RefEl::kTria(), 2},
    {lf::base::RefEl::kQuad(), 2}});
  // Fetch pointer to cell number 8
  const lf::mesh::Entity *cell_p = mesh_p->EntityByIndex(0, 8);
  // Output results of member functions of DofHandler object
  std::cout << "dofh.NumDofs() = " << dofh.NumDofs() << std::endl;
  std::cout << "dofh.NumLocalDofs(*cell_p) = " << dofh.NumLocalDofs(*cell_p)
  << std::endl;
  std::cout << "dofh.NumInteriorDofs(*cell_p) = "
  << dofh.NumInteriorDofs(*cell_p) << std::endl;
  std::cout << "dofh.GlobalDofIndices(*cell_p) = (";
  const auto gdof_idxs = dofh.GlobalDofIndices(*cell_p);
  for (lf::base::glb_idx_t idx : gdof_idxs) {
    std::cout << idx << ", ";
  }
  std::cout << ")" << std::endl;
  std::cout << "dofh.InteriorGlobalDofIndices(*cell_p) = (";
  const auto idof_idxs =  dofh.InteriorGlobalDofIndices(*cell_p);
  for (lf::base::glb_idx_t idx : idof_idxs) {
    std::cout << idx << ", ";
  }
  std::cout << ")" << std::endl;
}
```

Console output of test code:

```
dofh.NumDofs() = 57
dofh.NumLocalDofs(*cell_p) = 5
dofh.NumInteriorDofs(*cell_p) = 2
dofh.GlobalDofIndices(*cell_p) = (18, 11, 10, 43, 44, )
dofh.InteriorGlobalDofIndices(*cell_p) = (43, 44, )
```

▲

End Problem 0-3 , 8 pts.