R. Hiptmair
L. Scarabosio
E. Fonn

Spring Term 2013

Numerical Methods for Partial Differential Equations

ETH Zürich
D-MATH

Exam Winter 2014

Problem 0.1 Penalty Technique for Dirichlet Boundary Conditions [61 points]

Let $\Omega \subset \mathbb{R}^2$ be a bounded polygonal domain. For a parameter $\epsilon > 0$ consider the linear variational problem: seek $u_\epsilon \in H^1(\Omega)$

$$\int_\Omega \mathbf{grad}\, u_\epsilon \cdot \mathbf{grad}\, v \,\mathrm{d}\boldsymbol{x} + \epsilon^{-1} \int_{\partial\Omega} (u_\epsilon - g)\, v \,\mathrm{d}S = 0 \quad \forall v \in H^1(\Omega)\ , \tag{0.1.1}$$

where $g \in H^1(\partial\Omega)$ is a given continuous function on the boundary $\partial\Omega$.

(0.1a) [I, 5 points] Explain, why (0.1.1) has a unique solution for any g.

(0.1b) [I, 5 points] Find the strong form ("PDE-form") of the 2nd-order elliptic boundary value problem associated with the variational problem (0.1.1).

In the sequel, write $u^* \in H^1(\Omega)$ for the solution of

$$-\Delta u = 0 \quad \text{in } \Omega \quad , \quad u = g \quad \text{on } \partial\Omega\ . \tag{0.1.2}$$

(0.1c) [I, 10 points] Show that

$$\|\mathbf{grad}\, u_\epsilon\|^2_{L^2(\Omega)} + \epsilon^{-1}\|u_\epsilon - g\|^2_{L^2(\partial\Omega)} \le \|\mathbf{grad}\, u^*\|^2_{L^2(\Omega)}\ . \tag{0.1.3}$$

HINT: The solutions u_ϵ of (0.1.1) are minimizers of particular quadratic functionals on $H^1(\Omega)$.

(0.1d) [7 points] Give heuristic arguments why we can expect $u_\epsilon \to u^*$ for $\epsilon \to 0$.

Now we discretize (0.1.1) by means of linear Lagrangian finite elements $\mathcal{S}^0_1(\mathcal{M}) \subset H^1(\Omega)$ based on a triangular mesh of Ω. The usual nodal basis of $\mathcal{S}^0_1(\mathcal{M})$ composed of tent functions is used throughout, its ordering induced by the numbering of vertices.

(0.1e) [I, 10 points] Implement a LehrFEM-style MATLAB function

```
function Aloc = STIMA_Penal_LFE(Vertices,pp,bdflag)
```

that computes the element stiffness matrix $\mathbf{A}_K \in \mathbb{R}^{3,3}$ for the bilinear form of (0.1.1) on a triangle K. `Vertices` contains a 3×2-matrix whose rows provide the coordinates of the vertices of the triangle. `pp` passes the value of the **p**enalty **p**arameter $\epsilon > 0$, and the array `bdflags` of three

boolean values indicates whether the i^{th} edge of the triangle (the edge opposite the i^{th} vertex), $i = 1, 2, 3$, is located on $\partial\Omega$.

HINT: You may rely on the LehrFEM function `STIMA_Lapl_LFE` which computes the local stiffness matrix for the bilinear form associated with $-\Delta$ and linear Lagrangian finite elements.

(0.1f) [10 points] The given LehrFEM library function

```
function A = assemMat_LFE(Mesh,EHandle,varargin)
```

computes the stiffness matrix for the Galerkin finite element discretization of the Neumann problem

$$-\Delta u = 0 \quad \text{in } \Omega \quad , \quad \mathbf{grad}\, u \cdot \boldsymbol{n} = q \quad \text{on } \partial\Omega \,, \tag{0.1.4}$$

by means of linear Lagrangian finite elements on a triangular mesh of Ω.

Copy `assemMat_LFE.m` and modify it, using `STIMA_Penal_LFE` from subproblem (0.1e), to write a MATLAB assembly function

```
function A = assem_Penal_LFE(Mesh,pp)
```

that returns the stiffness matrix for the variational problem (0.1.1) for a Galerkin finite element discretization by means of $\mathcal{S}_1^0(\mathcal{M})$. The argument `Mesh` must supply a LehrFEM mesh data structure complete with edge information, while `pp` passes ϵ.
Assume that the structure `Mesh` contains the field `Mesh.BdFlags`, an array of length the number of edges in the mesh; if the edge `i` is on the boundary, then `Mesh.BdFlags(i)=-1`, if it is an inner edge, then `Mesh.BdFlags(i)=0`.

HINT:

- The `Vert2Edge`-field of `Mesh` can be used to obtain the index numbers of the edges of a triangle.
- A scrambled reference implementation of `STIMA_Penal_LFE` is available in the file `STIMA_Penal_LFE_ref.p`.

(0.1g) [7 points] The given LehrFEM function

```
function phi = assemLoad_Neu_LFE(Mesh,QHandle)
```

computes the right hand side vector for the Galerkin finite element discretization of the Neumann problem (0.1.4) by means of linear Lagrangian finite elements on a triangular mesh of Ω. The argument `GHandle` passes a function handle to the function $q : \partial\Omega \to \mathbb{R}$ that supplies the Neumann boundary values.

Use this function and your implementation of `assem_Penal_LFE` to devise a MATLAB function

```
function u_eps = solve_Penal_DirPrb(Mesh,GHandle,pp)
```

that computes the basis expansion coefficients of the $\mathcal{S}_1^0(\mathcal{M})$-finite element solution of (0.1.1). Here, `GHandle` passes a function handle of type `@(x)` to g.

HINT: A scrambled reference implementation of `assem_Penal_LFE` is provided in `assem_Penal_LFE_ref.p`.
For `assem_Penal_LFE`, again assume that the structure `Mesh.BdFlags` is already initialized.

(0.1h) [I, 7 points] We denote by $u_{\epsilon,i} \in \mathcal{S}_1^0(\mathcal{M}_i)$ the finite element Galerkin solutions of (0.1.1) on a sequence of triangular meshes $\mathcal{M}_0, \mathcal{M}_1, \ldots$ generated by regular uniform refinements. The penalty parameter ϵ is kept fixed.

In particular, we use as Ω the equilateral triangle with vertices $\binom{0}{0}$, $\binom{1}{0}$, and $\binom{\frac{1}{2}}{\frac{1}{2}\sqrt{3}}$, and $g(\boldsymbol{x}) := x_1^2 - x_2^2$. The doubly logarithmic plot of Figure 0.1 displays three lines. Which one correctly represents the semi-norm $|u_{\epsilon,i} - u^*|_{H^1(\Omega)}$, where u^* solves (0.1.2). Justify your answer.

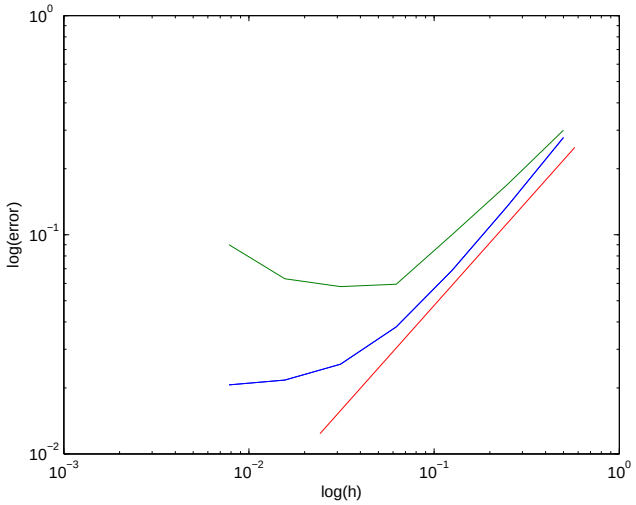

Figure 0.1: "Error curves" for subproblem (0.1h)

Listing 0.1: Testcalls for Problem 0.1

```
clear Mesh;
Mesh.Coordinates = [0 0; 1 0; 0.5 sqrt(3)/2];
Mesh.Elements = [1 2 3];
Mesh = add_Edges(Mesh);
GHandle = @(x,varargin) x(:,1).^2 - x(:,2).^2;
pp = 5e-3;
Loc = get_BdEdges(Mesh);
Mesh.BdFlags = zeros(size(Mesh.Edges,1),1);
Mesh.BdFlags(Loc) = -1;

fprintf('\n##STIMA_Penal_LFE');
Vertices = Mesh.Coordinates;
bdflag = [1 0 0];
STIMA_Penal_LFE(Vertices,pp,bdflag)

```

```
fprintf('\n##assem_Penal_LFE');
Mesh = refine_REG(Mesh);
assem_Penal_LFE(Mesh,pp)

fprintf('\n##solve_Penal_DirPrb');
solve_Penal_DirPrb(Mesh,GHandle,pp)
```

Listing 0.2: Output for Testcalls for Problem 0.1

```
>> test_call

##STIMA_Penal_LFE
ans =

    0.5774   -0.2887   -0.2887
   -0.2887   67.2440   33.0447
   -0.2887   33.0447   67.2440

##assem_Penal_LFE
ans =

   (1,1)       67.2440
   (4,1)       16.3780
   (5,1)       16.3780
   (2,2)       67.2440
   (4,2)       16.3780
   (6,2)       16.3780
   (3,3)       67.2440
   (5,3)       16.3780
   (6,3)       16.3780
   (1,4)       16.3780
   (2,4)       16.3780
   (4,4)       68.3987
   (5,4)       -0.5774
   (6,4)       -0.5774
   (1,5)       16.3780
   (3,5)       16.3780
   (4,5)       -0.5774
   (5,5)       68.3987
   (6,5)       -0.5774
   (2,6)       16.3780
   (3,6)       16.3780
   (4,6)       -0.5774
   (5,6)       -0.5774
   (6,6)       68.3987

##solve_Penal_DirPrb
ans =

   -0.0097
```

```
 0.9867
-0.4770
 0.1950
-0.0917
 0.3966
```

Problem 0.2 Thermal Evolution Problem [65 points]

On a 2D bounded polygonal spatial domain $\Omega \subset \mathbb{R}^2$ and time interval $[0, T]$, $T > 0$, we consider the following evolution problem in variational form: seek $t \mapsto u(t) \in H^1(\Omega)$ such that

$$\int_\Omega \frac{\partial u}{\partial t}(\boldsymbol{x}, t)\,\mathrm{d}\boldsymbol{x} \cdot \int_\Omega v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + \int_\Omega \mathbf{grad}\, u \cdot \mathbf{grad}\, v\,\mathrm{d}\boldsymbol{x} = 0 \quad \forall v \in H^1(\Omega)\ , \qquad u(\cdot, 0) = u_0 \quad \text{on } \Omega\ . \tag{0.2.1}$$

For the *spatial semi-discretization* of (0.2.1) we employ quadratic Lagrangian finite elements (space $\mathcal{S}_2^0(\mathcal{M})$) on a triangular mesh $\mathcal{M}$. We adopt the notation $\mathcal{V}(\mathcal{M})$ and $\mathcal{E}(\mathcal{M})$ for the sets of vertices and edges, respectively, of $\mathcal{M}$. As basis for $\mathcal{S}_2^0(\mathcal{M})$ we use the nodal basis associated with interpolation nodes located in the vertices and the midpoints of the edges of $\mathcal{M}$. We assume a numbering of vertices and edges of the mesh, which induces a numbering of the nodal basis functions. We follow the convention that in the ordered basis the vertex associated basis functions come before the edge associated.

Eventually, spatial semi-discretization leads to an ordinary differential equation (ODE) of the form

$$\mathbf{T}\dot{\vec{\boldsymbol{\mu}}} + \mathbf{A}\vec{\boldsymbol{\mu}} = 0\ , \tag{0.2.2}$$

for the vector $\vec{\boldsymbol{\mu}} = \vec{\boldsymbol{\mu}}(t)$, $0 \le t \le T$, of the finite element basis expansion coefficients of an approximation $u_N(t)$ of $u(t)$.

The *discretization in time* of (0.2.2) is done using the *2nd-order* implicit SDIRK-2 Runge-Kutta timestepping method, which is defined through the Butcher scheme

$$\begin{array}{c|cc} \lambda & \lambda & 0 \\ 1 & 1-\lambda & \lambda \\ \hline & 1-\lambda & \lambda \end{array}\ , \quad \lambda := 1 - \tfrac{1}{2}\sqrt{2}\ , \tag{0.2.3}$$

(0.2a) [I, 5 points] The evolution problem (0.2.1) can be written in the form

$$t \mapsto u(t) \in V: \quad \frac{\mathrm{d}}{\mathrm{d}t}\mathsf{m}(u(t), v) + \mathsf{a}(u, v) = 0 \quad \forall v \in V\ . \tag{0.2.4}$$

What are the space V and the bilinear forms m and a in the case of (0.2.1)?

(0.2b) [I, 7 points] Show that $\int_\Omega u(\boldsymbol{x}, t)\,\mathrm{d}\boldsymbol{x}$ does not change during the evolution (0.2.1).

(0.2c) [I, 5 points] What are the sizes of the matrices $\mathbf{M}$ and $\mathbf{A}$ in (0.2.2) in terms of $\sharp\mathcal{V}(\mathcal{M})$ and $\sharp\mathcal{E}(\mathcal{M})$?

HINT: The symbol $\sharp$ denotes the number of elements of a set.

(0.2d) [I, 7 points] Give a sharp upper bound for the number of possible non-zero entries in the matrix $\mathbf{A}$ in terms of $\sharp\mathcal{V}(\mathcal{M})$ and $\sharp\mathcal{E}(\mathcal{M})$.

(0.2e) [I, 10 points] The matrix $\mathbf{T}$ can be written in the form $\mathbf{T} = \mathbf{t}\mathbf{t}^\top$ with a suitable column vector $\mathbf{t}$. Describe the entries of $\mathbf{t}$.

HINT: It is useful to remember that the simple quadrature formula on a triangle K,

$$\int_K f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \approx \tfrac{1}{3}|K|\big(f(\boldsymbol{m}^1) + f(\boldsymbol{m}^2) + f(\boldsymbol{m}^3)\big)\,, \tag{0.2.5}$$

where the $\boldsymbol{m}^i$, $i = 1, 2, 3$, are the midpoints of the edges of K. This quadrature formula is *exact* for quadratic polynomials $f \in \mathcal{P}_2(K)$!

(0.2f) [I, 7 points] Show that $\mathbf{T} + \alpha\mathbf{A}$ is symmetric and positive definite for any $\alpha > 0$.

(0.2g) [10 points] Write an efficient LehrFEM-style MATLAB function

```
function tvec = assem_Tvec_QFE(Mesh)
```

that computes the vector $\mathbf{t}$ as introduced in subproblem (0.2e). Here, `Mesh` is a LehrFEM (triangular) mesh data structure complete with edge information.

HINT: Use the given function `area = ElemArea(Vertices)` which computes the area of the triangle with vertex coordinates contained in the 3×2-matrix `Vertices`.

(0.2h) [I, 7 points] Now we consider the fully discrete evolution. How do you have to adjust timestep in order to balance spatial and temporal errors, when one uniform regular refinement of the spatial mesh is performed and you are interested in the $H^1(\Omega)$-norm of the error? (Smoothness in space and time of the solution can be taken for granted).

(0.2i) [I, 7 points] Which linear systems of equations, expressed in terms of the matrices $\mathbf{T}$ and $\mathbf{A}$ have to be solved in every timestep of the fully discrete evolution?

Listing 0.3: Testcalls for Problem 0.2

```
Mesh.Coordinates = [0 0 ; 1 0; 0 1];
Mesh.Elements = [1 2 3];
Mesh = add_Edges(Mesh);
Mesh.BdFlags=zeros(length(Mesh.Edges),1);
Mesh = refine_REG(Mesh);
Mesh = add_Edge2Elem(Mesh);

fprintf('\n##assemTvecQFE');
t = assemTvecQFE(Mesh)
```

Listing 0.4: Output for Testcalls for Problem 0.2

```
>> t = assemTvecQFE(Mesh)

t =
```

```
4
5           0
6           0
7           0
8           0
9           0
10          0
11      0.0417
12      0.0417
13      0.0417
14      0.0417
15      0.0417
16      0.0417
17      0.0833
18      0.0833
19      0.0833
```

Problem 0.3 Transformed Convection-Diffusion Problem [76 points]

On a polygonal domain $\Omega \subset \mathbb{R}^2$ we consider two variational problems

$$u \in H_0^1(\Omega): \quad \underbrace{\int_\Omega (\mathbf{grad}\, u - \mathbf{grad}\, w\, u) \cdot \mathbf{grad}\, v \,\mathrm{d}\boldsymbol{x}}_{=:\mathsf{a}_1(u,v)} = \int_\Omega f v \,\mathrm{d}\boldsymbol{x} \quad \forall v \in H_0^1(\Omega)\,, \tag{0.3.1a}$$

$$\widetilde{u} \in H_0^1(\Omega): \quad \underbrace{\int_\Omega \exp(w(\boldsymbol{x}))\, \mathbf{grad}\, \widetilde{u} \cdot \mathbf{grad}\, v \,\mathrm{d}\boldsymbol{x}}_{=:\mathsf{a}_2(u,v)} = \int_\Omega f v \,\mathrm{d}\boldsymbol{x} \quad \forall v \in H_0^1(\Omega)\,, \tag{0.3.1b}$$

for $f \in L^2(\Omega)$.
Here, the given function w belongs to $\mathcal{C}^2(\overline{\Omega})$, that is, it is twice continuously differentiable on $\overline{\Omega}$ (up to the boundary).

(0.3a) [I, 5 points] Show that the bilinear form $\mathsf{a}_1(\cdot,\cdot)$ underlying the variational problem (0.3.1a) is *continuous* on $H^1(\Omega)$.

HINT: Remember that a bilinear form $\mathsf{a}(\cdot,\cdot)$ defined on a vector space V with norm $\|\cdot\|_V$ is called *continuous on* V, if and only if

$$|\mathsf{a}(u,v)| \le C\|u\|_V\|v\|_V \quad \forall u, v \in V\,, \tag{0.3.2}$$

for some $C > 0$.

(0.3b) [I, 5 points] State the 2nd-order boundary value problem satisfied by the solution $\widetilde{u}$ of (0.3.1b).

(0.3c) [I, 10 points] Which 2nd-order boundary value problem is solved by u from (0.3.1a)?

(0.3d) [I, 7 points] Show existence and uniqueness of solutions of (0.3.1b).

HINT: You may cite theoretical results presented in the course.

(0.3e) [I, 7 points] Show that the function $\boldsymbol{x} \mapsto \exp(-w(\boldsymbol{x}))v(\boldsymbol{x})$ belongs to $H^1(\Omega)$, if $v \in H^1(\Omega)$.

HINT: Some of the following product rule formulas from vector analysis may be be useful:

$$\mathbf{grad}(uv) = \mathbf{grad}\, u \cdot v + u\, \mathbf{grad}\, v\ , \quad u, v \in H^1(\Omega)\ , \tag{0.3.3}$$

$$\operatorname{div}(\mathbf{w}u) = \operatorname{div} \mathbf{w}\, u + \mathbf{w} \cdot \mathbf{grad}\, u\ , \tag{0.3.4}$$

$$\operatorname{div}(\mathbf{grad}\, u) = \Delta u\ . \tag{0.3.5}$$

(0.3f) [I, 10 points] Show that $\widetilde{u}(\boldsymbol{x}) = \exp(-w(\boldsymbol{x}))u(\boldsymbol{x})$, $\boldsymbol{x} \in \Omega$, where u solves (0.3.1a) and $\widetilde{u}$ solves (0.3.1b).

HINT: Use the hint given for (0.3e).

Let Ω be equipped with a triangular mesh $\mathcal{M}$. The function w in 0.3.1 is now approximated by a piecewise linear finite element function: $w_N \in \mathcal{S}_1^0(\mathcal{M})$. We perform a Ritz-Galerkin discretization of both (0.3.1a) and (0.3.1b) (with w replaced by w_N) based on the trial and test space $\mathcal{S}_{1,0}^0(\mathcal{M})$ of linear Lagrangian finite element functions on $\mathcal{M}$ that vanish on the boundary $\partial\Omega$. Let u_N and $\widetilde{u}_N$ denote the resulting approximate solutions of (0.3.1a) and (0.3.1b), respectively.

(0.3g) [I, 5 points] Argue, why, in contrast to what we found in subproblem (0.3f), the relationship $\widetilde{u}_N(\boldsymbol{x}) = \exp(-w_N(\boldsymbol{x}))u_N(\boldsymbol{x})$, $\boldsymbol{x} \in \Omega$, does not hold in general.

(0.3h) [I, 10 points] Denote by $\mathbf{A}_K^{\Delta}$ the element stiffness matrix for the bilinear form associated with $-\Delta$ and linear Lagrangian finite elements for a triangle K, and by $\boldsymbol{\mu}_w \in \mathbb{R}^3$ the vector containing the values of w_N at the three vertices of K.
Write the element stiffness matrix $\mathbf{A}_K^1 \in \mathbb{R}^{3,3}$ on the triangle $K \in \mathcal{M}$ for the variational problem (0.3.1a) (with w replaced by w_N) and linear Lagrangian finite elements in terms of $\mathbf{A}_K^{\Delta}$ and $\boldsymbol{\mu}_w$.

HINT: For a linear function f on the triangle K with vertices $\boldsymbol{a}^1$, $\boldsymbol{a}^2$, $\boldsymbol{a}^3$, it holds:

$$\int_K f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \tfrac{1}{3}|K|\big(f(\boldsymbol{a}^1) + f(\boldsymbol{a}^2) + f(\boldsymbol{a}^3)\big)\ . \tag{0.3.6}$$

(0.3i) [5 points] Implement a MATLAB function

```
function Aloc = STIMA_VP1_LFE(Vertices,wvals)
```

that computes the element stiffness matrix $\mathbf{A}_K^1 \in \mathbb{R}^{3,3}$ on the triangle $K \in \mathcal{M}$ for the variational problem (0.3.1a) (with w replaced by w_N) and linear Lagrangian finite elements. Following the LehrFEM conventions, the argument `Vertices` is a 3×2-matrix, whose rows contain the coordinates of the vertices of K. The argument `wvals` is a column vector of length 3, whose entries provide the values of $w_N \in \mathcal{S}_1^0(\mathcal{M})$ in the vertices of K (i.e. `wvals` coincides with the vector $\boldsymbol{\mu}_w$ of subproblem (0.3h)). The local numbering of the vertices follows their order in `Vertices`.

For the implementation you may rely on the LehrFEM library function `Aloc = STIMA_Lapl_LFE(Vertices)` that computes the local stiffness matrix $\mathbf{A}_K^{\Delta}$ for the bilinear form associated with $-\Delta$ and linear Lagrangian finite elements.

(0.3j) **[I, 7 points]** Denote again by $\mathbf{A}_K^{\Delta}$ the element stiffness matrix for the bilinear form associated with $-\Delta$ and linear Lagrangian finite elements for a triangle K.
Write the element stiffness matrix $\mathbf{A}_K^2 \in \mathbb{R}^{3,3}$ on the triangle $K \in \mathcal{M}$ for the variational problem (0.3.1b) (with w replaced by w_N) and linear Lagrangian finite elements in terms of $\mathbf{A}_K^{\Delta}$ and $\boldsymbol{\mu}_w$.

To evaluate integrals of continuous functions over a triangle K use the quadrature formula

$$\int_K f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \approx \tfrac{1}{3}|K|\big(f(\boldsymbol{m}^1)+f(\boldsymbol{m}^2)+f(\boldsymbol{m}^3)\big)\;, \tag{0.3.7}$$

where $\boldsymbol{m}^1$, $\boldsymbol{m}^2$, and $\boldsymbol{m}^3$ are the midpoints of the edges of K.

(0.3k) **[5 points]** Implement a MATLAB function

```
function Aloc = STIMA_VP2_LFE(Vertices,wvals)
```

that provides the element stiffness matrices for (0.3.1b) (with w replaced by w_N) and its Ritz-Galerkin discretization on the finite element space $\mathcal{S}^0_{1,0}(\mathcal{M})$. The arguments are the same as those for `STIMA_VP1_LFE` from subproblem (0.3i).

For the implementation you may again rely on the LehrFEM library function `Aloc = STIMA_Lapl_LFE(Vertices` that computes the local stiffness matrix $\mathbf{A}_K^{\Delta}$ for the bilinear form associated with $-\Delta$ and linear Lagrangian finite elements.

Listing 0.5: Testcalls for Problem 0.3

```
fprintf('##STIMA_VP1_LFE')
Vertices=[0 0;1 0; 0 1];
wvals=[1 2 3]';
A1 = STIMA_VP1_LFE(Vertices,wvals)

fprintf('##STIMA_VP2_LFE')
A2 = STIMA_VP2_LFE(Vertices,wvals)
```

Listing 0.6: Output for Testcalls for Problem 0.3

```
>> test_call
##STIMA_VP1_LFE
A1 =

    1.5000         0         0
   -0.6667    0.3333   -0.1667
   -0.8333   -0.3333    0.1667

##STIMA_VP2_LFE
A2 =

    1.6348   -0.8174   -0.8174
   -0.8174    0.8174    0.0000
   -0.8174    0.0000    0.8174
```

Problem 0.4 Averaging Recovery of Continuous Gradients [84 points]

The polygonal bounded domain $\Omega \subset \mathbb{R}^2$ is equipped with a triangular mesh $\mathcal{M}$. A numbering of the triangles and vertices of the mesh is taken for granted. Let us denote by $\mathcal{V}(\mathcal{M})$ the set of vertices of $\mathcal{M}$.

We write $\mathcal{S}_1^0(\mathcal{M})$ for the space of $\mathcal{M}$-piecewise linear continuous functions, and $\mathcal{S}_0^{-1}(\mathcal{M})$ for the space of $\mathcal{M}$-piecewise constant functions. These spaces are equipped with nodal bases; (i) as basis for $\mathcal{S}_1^0(\mathcal{M})$ we use the locally supported tent functions ordered according to the numbering of the vertices of the mesh, and (ii) as basis for $\mathcal{S}_0^{-1}(\mathcal{M})$ we rely on the characteristic functions of the triangles of the mesh, that is, functions that attain the value 1 on a particular triangle and vanish on all others. Their numbering is induced by that of the triangles.

For each vertex $\boldsymbol{p} \in \mathcal{V}(\mathcal{M})$ of the mesh, define $\mathcal{M}_{\boldsymbol{p}}$ as the set of triangles that have $\boldsymbol{p}$ as a vertex, write $\omega_{\boldsymbol{p}} \subset \overline{\Omega}$ for the union (of the closures) of these triangles, and $|\omega_{\boldsymbol{p}}|$ for its area.

The so-called gradient recovery operator

$$\mathsf{R} : \mathcal{S}_1^0(\mathcal{M}) \to (\mathcal{S}_1^0(\mathcal{M}))^2 \tag{0.4.1a}$$

is defined by

$$(\mathsf{R}u_N)(\boldsymbol{p}) := \frac{1}{|\omega_{\boldsymbol{p}}|} \sum_{K \in \mathcal{M}_{\boldsymbol{p}}} |K| (\mathbf{grad}\, u_N\big|_K)(\boldsymbol{p}) \quad \text{for every } \boldsymbol{p} \in \mathcal{V}(\mathcal{M}). \tag{0.4.1b}$$

Here, $\mathbf{grad}\, u_N\big|_K$ refers to the value of $\mathbf{grad}\, u_N$ on the triangle K.

(0.4a) [I, 5 points] Explain, why (0.4.1a)–(0.4.1b) is a valid definition of R.

(0.4b) [I, 12 points] Show that for all $u_N \in \mathcal{S}_1^0(\mathcal{M})$

$$\int_\Omega (\mathsf{R}u_N)(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_\Omega \mathbf{grad}\, u_N(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.$$

HINT: For $f \in \mathcal{P}_1(K)$ it holds:

$$\int_K f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \tfrac{1}{3}|K|\big(f(\boldsymbol{a}^1) + f(\boldsymbol{a}^2) + f(\boldsymbol{a}^3)\big)\ . \tag{0.4.2}$$

with $\boldsymbol{a}^1$, $\boldsymbol{a}^2$ and $\boldsymbol{a}^3$ the vertices of the triangle K.

(0.4c) [I, 5 points] You read the statement

> R *linearly* (a) maps *linear* (b) Lagrangian finite element functions into piecewise *linear* (c) vector-fields.

Explain the meanings of "linear" at occurrences (a), (b), and (c).

(0.4d) [I, 10 points] Compute the matrix representation of $\mathbf{grad} : \mathcal{S}_1^0(\mathcal{M}) \to (\mathcal{S}_0^{-1}(\mathcal{M}))^2$ for the mesh displayed in Figure 0.2 using the vertex and cell numbering given there. Use the following ordered basis of $(\mathcal{S}_0^{-1}(\mathcal{M}))^2$:

$$\left\{ \begin{pmatrix} q_1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ q_1 \end{pmatrix}, \begin{pmatrix} q_2 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ q_2 \end{pmatrix} \right\}, \tag{0.4.3}$$

where q_i is the characteristic function of the i^{th} triangle.

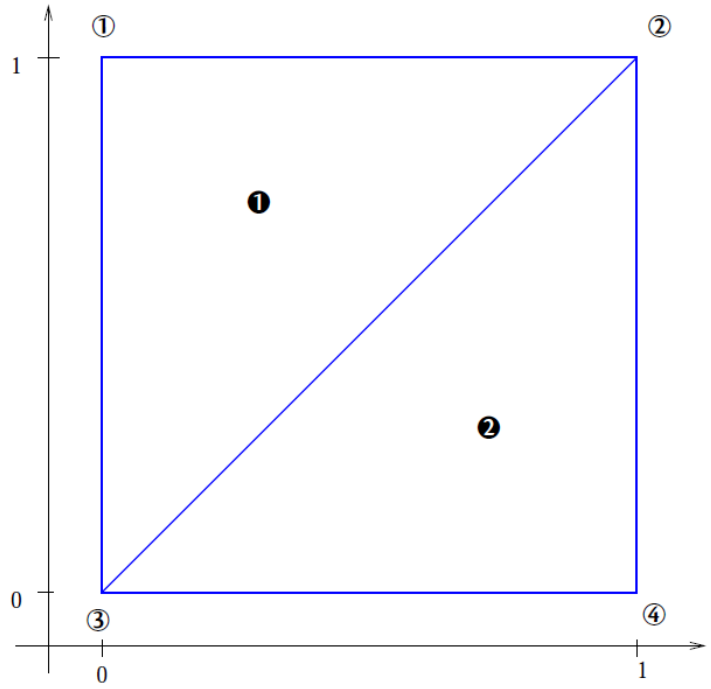

Figure 0.2: Simple triangular mesh to be used in subproblems (0.4d) and (0.4e): ①–④ give the numbers of vertices, ❶, ❷ the numbers of triangles.

(0.4e) [I, 10 points] Let $\mathsf{R} \in \mathbb{R}^{8,4}$ denote the matrix representation of R for the simple mesh from Figure 0.2. As basis for $(\mathcal{S}_1^0(\mathcal{M}))^2$ use

$$\left\{ \begin{pmatrix} b_N^1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^1 \end{pmatrix}, \begin{pmatrix} b_N^2 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^2 \end{pmatrix} \begin{pmatrix} b_N^3 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^3 \end{pmatrix}, \begin{pmatrix} b_N^4 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^4 \end{pmatrix} \right\}, \tag{0.4.4}$$

where b_N^i is the tent function associated with the i^{th} vertex of the mesh, see Figure 0.2 for the numbering. Compute the first column of R.

(0.4f) [I, 12 points] Implement an *efficient* LehrFEM-style MATLAB function

```
function parea = get_Vertex_Region_Areas(Mesh)
```

that expects a *simple* LehrFEM mesh data structure (`Mesh.Coordinates` and `Mesh.Elements` only) in the argument `Mesh` and returns a row vector of size N, $N := \sharp\mathcal{V}(\mathcal{M})$, whose i^{th} entry contains $|\omega_{\boldsymbol{p}}|$, if $\boldsymbol{p} \in \mathcal{V}(\mathcal{M})$ is the i^{th} vertex of the mesh.

HINT: Use the given function `area = ElemArea(Vertices)` which computes the area of the triangle with vertex coordinates contained in the 3×2-matrix `Vertices`.
The symbol $\sharp$ denotes the number of elements of a set.

(0.4g) [20 points] Write an *efficient* LehrFEM-style MATLAB function

```
function ug = avg_gradient(Mesh,u)
```

that returns the $2N$-vector, $N := \sharp\mathcal{V}(\mathcal{M})$, of basis expansion coefficients for $\mathsf{R}u_N$ in `ug`, when `Mesh` passes a simple LehrFEM mesh data structure, and $u_N \in \mathbb{R}^N$ the coefficient vector of $u_N \in \mathcal{S}_1^0(\mathcal{M})$. As basis for $(\mathcal{S}_1^0(\mathcal{M}))^2$ we use

$$\left\{ \begin{pmatrix} b_N^1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^1 \end{pmatrix}, \begin{pmatrix} b_N^2 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^2 \end{pmatrix}, \dots \begin{pmatrix} b_N^{N-1} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^{N-1} \end{pmatrix}, \begin{pmatrix} b_N^N \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ b_N^N \end{pmatrix} \right\}, \tag{0.4.5}$$

where b_N^i is the tent function associated with vertex i.

HINT: You can use the LehrFEM function `grad_shap_LFE` to evaluate the gradients of the local shape functions *on the reference element*.

(0.4h) [I, 10 points] We consider a sequence $\mathcal{M}_i$, $i \in \mathbb{N}_0$, of triangular meshes of Ω obtained by successive uniform regular refinement of an initial mesh $\mathcal{M}_0$. Denote by I_i for the nodal interpolation operator $\mathsf{I}_i : C^0(\overline{\Omega}) \to \mathcal{S}_1^0(\mathcal{M}_i)$. Further, write $\mathsf{R}_i : \mathcal{S}_1^0(\mathcal{M}_i) \to (\mathcal{S}_1^0(\mathcal{M}_i))^2$ for the gradient averaging operator according to (0.4.1b) on $\mathcal{M}_i$.

For Ω the equilateral triangle with vertices $\binom{0}{0}$, $\binom{1}{0}$, and $\binom{\frac{1}{2}}{\frac{1}{2}\sqrt{3}}$ and $f(\boldsymbol{x}) := \exp(\|\boldsymbol{x}\|^2)$, Table 0.1 lists the "errors" $\|\mathbf{grad}\,\mathsf{I}_i f - \mathbf{grad}\, f\|_{L^2(\Omega)}$ and $\|\mathsf{R}_i\mathsf{I}_i f - \mathbf{grad}\, f\|_{L^2(\Omega)}$ as functions of the meshwidth of $\mathcal{M}_i$ for some meshes of the family.

Describe qualitatively and quantitatively the convergence of the "error norms" in terms of the meshwidth h that can be concluded from the experimental data.

HINT: The data of Table 0.1 are available as the MATLAB file `tab.dat`, to be loaded with the command `load('tab.mat')`. The variable `tab.dat` is a struct with three fields: `tab.h` contains the meshwidth data, `tab.err1` contains the errors $\|\mathbf{grad}\,\mathsf{I}_i f - \mathbf{grad}\, f\|_{L^2(\Omega)}$ and `tab.err2` contains the errors $\|\mathsf{R}_i\mathsf{I}_i f - \mathbf{grad}\, f\|_{L^2(\Omega)}$.

mesh	meshwidth h	$\|\mathbf{grad}\,\mathsf{I}_i f - \mathbf{grad}\, f\|_{L^2(\Omega)}$	$\|\mathsf{R}_i\mathsf{I}_i f - \mathbf{grad}\, f\|_{L^2(\Omega)}$
$\mathcal{M}_1$	0.1250	0.1337	0.0975
$\mathcal{M}_2$	0.0625	0.0671	0.0380
$\mathcal{M}_3$	0.0313	0.0336	0.0141
$\mathcal{M}_4$	0.0156	0.0168	0.0051
$\mathcal{M}_5$	0.0078	0.0084	0.0018
$\mathcal{M}_6$	0.0039	0.0042	0.0007

Table 0.1: Measured "errors" for (recovered) gradients of piecewise linear interpolants

Listing 0.7: Testcalls for Problem 0.4

```
clear Mesh;

Mesh.Coordinates = [0 0; 1 0; 0.5 sqrt(3)/2];
Mesh.Elements = [1 2 3];

Mesh = add_Edges(Mesh);
Loc = get_BdEdges(Mesh);
Mesh.BdFlags = zeros(size(Mesh.Edges,1),1);
Mesh.BdFlags(Loc) = -1;
```

```
Mesh.ElemFlag = ones(size(Mesh.Elements,1),1);
Mesh = add_Edge2Elem(Mesh);
Mesh = refine_REG(Mesh);
Mesh = refine_REG(Mesh);

F = @(x) exp(x(:,1).^2+x(:,2).^2);

fprintf('\n##get_Vertex_Region_Areas');
areas = get_Vertex_Region_Areas(Mesh)

u = F(Mesh.Coordinates);
fprintf('\n##avg_gradient');
ug = avg_gradient(Mesh,u)
```

Listing 0.8: Output for Testcalls for Problem 0.4

```
test_calls

>> test_call

##get_Vertex_Region_Areas14  end

areas =

  Columns 1 through 10

    0.0271    0.0271    0.0271    0.0812    0.0812    0.0812
       0.0812    0.0812    0.0812    0.0812

  Columns 11 through 15

    0.0812    0.0812    0.1624    0.1624    0.1624

##avg_gradient
ug =

    0.2580
    0.1489
    3.8529
    0.0779
    1.9939
    3.2978
    1.3775
    0.2383
    0.8951
    1.0739
    2.4547
    1.4172
    0.5677
    0.2080
```

```
    0.4640
    0.3876
    3.1635
    0.5832
    2.8519
    0.2960
    1.6824
    2.3218
    2.0868
    2.4481
    0.9687
    0.5593
    2.0892
    0.7237
    1.6714
    1.4475
```

References

[NPDE] Lecture Slides for the course “Numerical Methods for Partial Differential Equations”, SVN revision # 62366.

[NCSE] Lecture Slides for the course “Numerical Methods for CSE”.

[LehrFEM] LehrFEM manual.