R. Hiptmair
L. Scarabosio
E. Fonn

Spring Term 2013

Numerical Methods for Partial Differential Equations

ETH Zürich
D-MATH

Exam Summer 2013

Problem 1 Entropy Solution [19 points]

We consider the Cauchy problem for the 1D scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(\cosh(u)) = 0 \quad \text{in } \mathbb{R}\times]0,T[\,, \tag{1.1}$$

$$u(x,0) = u_0(x) \quad x \in \mathbb{R}\,, \tag{1.2}$$

where $\cosh(u) = \frac{1}{2}(e^x + e^{-x})$ with $\cosh'(u) = \sinh(u) := \frac{1}{2}(e^x - e^{-x})$, see Figure 1.1

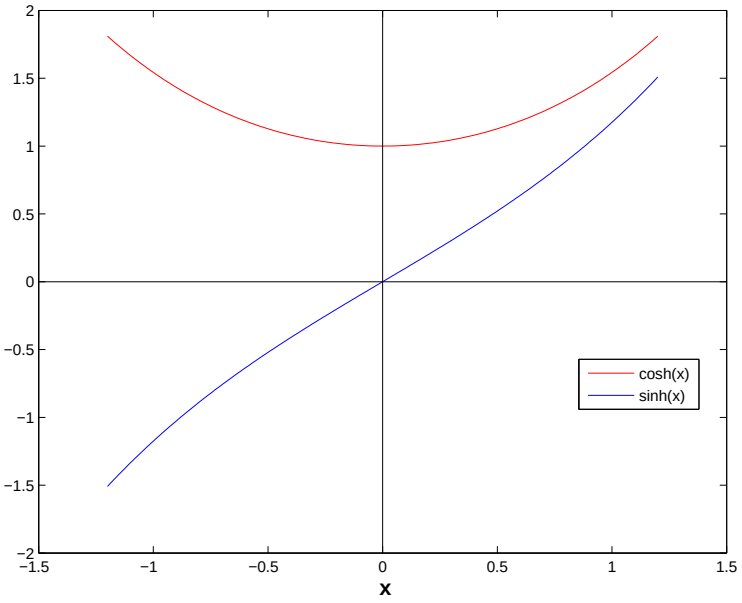

Figure 1.1: Graphs of hyperbolic trigonometric functions

(1a) [I, 3 points]

Show that for

$$u_0(x) = \begin{cases} -1 & \text{, for } x < 0 \\ 1 & \text{, for } x > 0\,, \end{cases} \tag{1.3}$$

$u(x,t) = u_0(x)$ for all $t > 0$, $x \in \mathbb{R}$, is a valid weak solution.

(1b) [I, 3 points]

Explain why the weak solution (1.3) fails to be an entropy solution?

(1c) [I, 3 points]

In the $x-t$-plane sketch the zone, where the entropy solution of (1.1) with initial data (1.3) can be different from ± 1.

(1d) [I, 4 points]

Sketch the entropy solution of (1.1) with initial data (1.3) at time $t = 1$.

(1e) [I, 6 points]

Implement a MATLAB function

```
function F = godnfn(v,w)
```

that implements the *Godunov numerical flux function* $F_{\mathrm{GD}}(v, w)$ from [NPDE, Equ. 8.3.35] for use with a conservative finite volume method for (1.1).

Problem 2 Crouzeix-Raviart Finite Elements [60 points]

Let a triangular mesh $\mathcal{M}$ of a 2D polygonal bounded domain $\Omega \subset \mathbb{R}^2$ be given and write $\mathcal{N} = \{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_N\}$ for the set of the midpoints of its edges. A numbering of these points is assumed.

The so-called Crouzeix-Raviart finite element space $\mathcal{CR}(\mathcal{M}) \subset L^2(\Omega)$ on the mesh $\mathcal{M}$ is defined as the span of the functions b_N^j, $j = 1, \ldots, N$, which satisfy

$$b_{N\,|K}^i \in \mathcal{P}_1(K) \quad \forall K \in \mathcal{M} \quad , \quad b_N^i(\boldsymbol{m}_j) = \begin{cases} 1 & \text{, if } i = j \text{ ,} \\ 0 & \text{else,} \end{cases} \quad i, j \in \{1, \ldots, N\} \text{ .} \tag{2.1}$$

(2a) [I, 2 points]

Show that (2.1) provides a valid definition of the functions b_N^j.

(2b) [I, 2 points]

Show that the set of functions $\{b_N^j : j = 1, \ldots, N\}$ is linearly independent.

(2c) [I, 3 points]

Show that $\mathcal{CR}(\mathcal{M}) \not\subset H^1(\Omega)$.

(2d) [I, 2 points]

Describe the support of a basis function b_N^i.

(2e) [I, 3 points]

We use the b_N^i from (2.1) as global shape functions. Show that the local shape functions for $\mathcal{CR}(\mathcal{M})$ on a triangle K can be expressed in terms of the barycentric coordinate functions λ_i on K as follows:

$$b_{N\,|K}^j = 1 - 2\lambda_{\mathrm{opp}(j)} \text{ ,} \quad j = 1 \ldots, N \text{ ,} \tag{2.2}$$

where $\mathrm{opp}(j)$ is the local index of the vertex opposite of $\boldsymbol{m}_j$ on triangle $K \in \mathcal{M}$.

(2f) [I, 4 points]

Compute the element (Galerkin) matrix for the finite element space $\mathcal{CR}(\mathcal{M})$ and the bilinear form

$$\mathsf{a}(u,v) = \int_\Omega uv \,\mathrm{d}\boldsymbol{x} \;, \quad u,v \in L^2(\Omega) \;.$$

HINT: Use the integral formula for barycentric coordinate functions on a triangle K:

$$\int_K \lambda_1^{\alpha_1}\lambda_2^{\alpha_2}\lambda_3^{\alpha_3} \,\mathrm{d}\boldsymbol{x} = |K| \frac{\alpha_1!\alpha_2!\alpha_3!2!}{(\alpha_1+\alpha_2+\alpha_3+2)!} \;.$$

For the implementation of finite element methods based on Crouzeix-Raviart finite element spaces in LehrFEM we use the b_N^j from (2.1) as global shape functions and number them according the intrinsic numbering of edges in LehrFEM as induced by the `Edges` field of the mesh data structure.

(2g) [6 points]

Implement a LehFEM function

```
function l2err = L2Err_CR(Mesh,mu,Fhandle)
```

that takes an extended LehrFEM mesh data structure complete with edge information (in `Mesh`), a coefficient vector `mu` of length $\mathcal{N}$ describing a function $u_N \in \mathcal{CR}(\mathcal{M})$ (in `mu`), and a handle of type `@(x)` to a continuous function $u : \Omega \mapsto \mathbb{R}$ (in `FHandle`). The return value should provide an approximation for $\|u - u_N\|_{L^2(\Omega)}$ computed by means of the local numerical quadrature offered by the LehrFEM function `P7O6`.

HINT: You can use the LehrFEM function `shap_LFE` to get values of the barycentric coordinate functions on the reference element. Use the Mesh data field `Vert2Edge` to access edge numbers.

(2h) [I, 3 points]

We introduce the mesh-dependent bilinear form

$$\mathsf{a}_{\mathcal{M}}(u,v) := \sum_{K\in\mathcal{M}} \int_K \mathbf{grad}\, u \cdot \mathbf{grad}\, v \,\mathrm{d}\boldsymbol{x} \;. \tag{2.3}$$

Derive a formula for the entries of the element matrices for the Galerkin discretization of $\mathsf{a}_{\mathcal{M}}$ based on $\mathcal{CR}(\mathcal{M})$. The entries of the element matrix for triangle K should be expressed in terms of the angles ω_k, $k = 1,2,3$ of K, see [NPDE, Figure 117]. Follow the convention that the i-th local edge is opposite to the i-th local vertex, $i = 1,2,3$.

HINT: Use (2.2) and [NPDE, Formula 3.2.10].

(2i) [2 points]

Implement the LehrFEM function

```
function Aloc = STIMA_Lapl_CR(Vertices)
```

that computes the element matrix for $\mathsf{a}_{\mathcal{M}}$ and the Crouzeix-Raviart finite element space. The local numbering scheme of the previous sub-problem applies and the 3×2 matrix `Vertices` contains the coordinates of the vertices of a triangle in its rows.

HINT: Recall the result of the sub-problem (2h) and use the LehrFEM function `Aloc = STIMA_Lapl_LFE(Vertices,varargin)` from the LehrFEM library that computes the element matrices for $\mathsf{a}_{\mathcal{M}}$ for the piecewise linear Lagrangian finite element spaces $\mathcal{S}_1^0(\mathcal{M})$.

(2j) [I, 6 points]

Implement an *efficient* LehrFEM function

```
function A = assemMat_Lapl_CR(Mesh)
```

that computes the (global) Galerkin matrix for the bilinear form $\mathsf{a}_{\mathcal{M}}$ discretized by means of a Crouzeix Raviart finite element space defined on a triangular mesh passed through the LehrFEM mesh data structure `Mesh`. You can take for granted that complete edge information is available in `Mesh`. The choice of global shape functions and numbering schemes as introduced above still apply.

HINT: You may rely on the function `STIMA_Lapl_CR` from sub-problem (2i). The field `Vert2Edge` of the extended LehrFEM mesh data structure comes handy.

(2k) [I, 2 points]

Prove that the Galerkin matrices for $\mathsf{a}_{\mathcal{M}}$ and the finite element spaces $\mathcal{CR}(\mathcal{M})$ are always positive semidefinite.

(2l) [I, 6 points]

Assume that Ω is connected. Show that the kernel of the Galerkin matrix arising from the discretization of $\mathsf{a}_{\mathcal{M}}$ based on the basis $\{b_N^j\}_{j=1}^N$ of $\mathcal{CR}(\mathcal{M})$ from (2.1) is one-dimensional and spanned by the vector with all entries $= 1$.

(2m) [I, 6 points]

Given a triangular mesh $\mathcal{M}$ of Ω and a continuous function $g : \partial\Omega \mapsto \mathbb{R}$, we consider the following discrete variational problem: seek $u_N \in \mathcal{CR}(\mathcal{M})$ such that

$$u_N(\boldsymbol{m}) = g(\boldsymbol{m}) \quad \forall \boldsymbol{m} \in \mathcal{N}_\partial \quad , \quad \mathsf{a}_{\mathcal{M}}(u_N, v_N) = 0 \quad \forall v_N \in \mathcal{CR}_0(\mathcal{M}) \; . \tag{2.4}$$

Here the following notations have been used:

- $\mathcal{N}_\partial := \{\boldsymbol{p} \in \mathcal{N} : \; \boldsymbol{p} \in \partial\Omega\}$ (midpoints of edges on the boundary) ,
- $\mathcal{CR}_0(\mathcal{M}) := \{v \in \mathcal{CR}(\mathcal{M}) : \; v(\boldsymbol{m}) = 0 \; \forall \boldsymbol{m} \in \mathcal{N}_\partial\}$.

Implement a LehrFEM function

```
function mu = Solve_LaplDir_CR(Mesh, GHandle)
```

that computes the coefficient vector (w.r.t. the basis $\left\{b_N^j\right\}_{j=1}^N$ from (2.1)) for the solution $u_N \in \mathcal{CR}(\mathcal{M})$ of (2.4). The argument `Mesh` passes a *basic* LehrFEM mesh data structure for $\mathcal{M}$ (with fields `Coordinates` and `Elements` initialized) and `GHandle` is a function handle of type `@(x)` providing the function g. The returned coefficient vector for u_N should have N components, where N is the number of *all* edges in the mesh $\mathcal{M}$.

HINT: Perform a treatment of Dirichlet boundary conditions by elimination as explained in [NPDE, Example 3.5.57] and [NPDE, Example 3.5.61].

(2n) [I, 4 points]

Somebody claims that the Crouzeix-Raviart finite element space and, in particular, the LehrFEM function `Solve_LaplDir_CR` from sub-problem (2m) can be used to solve the boundary value problem

$$-\Delta u = 0 \quad \text{in } \Omega, \quad u = g \quad \text{on } \partial\Omega\,, \tag{2.5}$$

though $\mathcal{CR}(\mathcal{M}) \not\subset H^1(\Omega)$! Test this claim numerically for $\Omega =]0,1[^2$ by using $u(\boldsymbol{x}) = \log(\left\|\boldsymbol{x} + \binom{1}{0}\right\|)$, which satisfies $\Delta u = 0$ on Ω, and computing $\|u - u_N\|_{L^2(\Omega)}$ for four different meshes arising from global regular refinement of an initial coarse mesh read in from the supplied files `Coords.dat` and `Elems.dat`.

To that end complete the implementation of the MATLAB function

```
[l2errvec,meshwidths] = cvg_LaplDir_CR
```

so that it returns a vector of the values $\|u - u_N\|_{L^2(\Omega)}$ and a vector of mesh-widths of the corresponding meshes.

HINT: The return values you should get can be loaded from the MATLAB data file `l2err_CR.dat`.

(2o) [I, 3 points]

Based on the numerical results of the previous sub-problem, describe qualitatively and quantitatively the observed convergence of $\|u - u_N\|_{L^2(\Omega)}$ as the mesh-width tends to 0.

HINT: The error norms and corresponding mesh widths are also available in the MATLAB data file `l2err_CR.dat`, which can be read by MATLAB's `load` function.

(2p) [I, 6 points]

Describe under which assumptions on the mesh $\mathcal{M}$ the solution of (2.4) will satisfy a special discrete maximum principle in the sense that

$$\min_{\boldsymbol{p}\in\mathcal{N}_\partial} u_N(\boldsymbol{p}) \le u_N(\boldsymbol{m}) \le \max_{\boldsymbol{p}\in\mathcal{N}_\partial} u_N(\boldsymbol{p}) \quad \forall \boldsymbol{m} \in \mathcal{N}\,. \tag{2.6}$$

HINT: Use the result from subproblem (2h).

Problem 3 2D Finite Volume Method for Advection [39 points]

For a continuous velocity field $\mathbf{a} : \Omega \mapsto \mathbb{R}^2$, on $\Omega =]-R, R[^2$ we consider the linear advection problem

$$\frac{\partial u}{\partial t} + \operatorname{div}_{\boldsymbol{x}}(\mathbf{a}(\boldsymbol{x})u) = 0 \quad \text{in } \Omega \times]0, T[\,, \tag{3.1}$$

$$u(\boldsymbol{x}, 0) = u_0(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega \,, \quad u(\boldsymbol{x}, t) = 0 \quad \forall 0 \le t \le T \,, \quad \boldsymbol{x} \in \Gamma_{\text{in}} \,, \tag{3.2}$$

with prescribed initial data $u_0 \in C^0(\overline{\Omega})$, and inflow boundary

$$\Gamma_{\text{in}} := \{\boldsymbol{x} \in \partial\Omega : \, \boldsymbol{n}(\boldsymbol{x}) \cdot \mathbf{a}(\boldsymbol{x}) < 0\} \,,$$

where $\boldsymbol{n}$ is the exterior unit normal vector field on $\partial\Omega$.

Throughout this problem we assume $\|\mathbf{a}\| \le 1$, where $\|\cdot\|$ designates the Euclidean norm of a vector.

For the sake of discretization Ω will be equipped with a triangular mesh $\mathcal{M} = \{K_1, K_2, \ldots, K_N\}$, whose set of edges will be denoted by $\mathcal{E}$. An approximation of the solution u is sought in the space $\mathcal{S}_0^{-1}(\mathcal{M})$ of piecewise constant functions on $\mathcal{M}$. Throughout, the characteristic functions of the mesh cells,

$$b_N^j(\boldsymbol{x}) = \begin{cases} 1 & \text{, if } \boldsymbol{x} \in K_j \,, \\ 0 & \text{elsewhere,} \end{cases} \tag{3.3}$$

will serve as global shape functions. Their numbering will be induced by the numbering of the mesh cells.

(3a) [I, 2 points]

Show that every continuous classical solution of (3.1) satisfies

$$\sum_{K \in \mathcal{M}} \left\{ \int_K \frac{\partial u}{\partial t}(\boldsymbol{x}, t) w_N(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + \int_{\partial K} \mathbf{a}(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x}) \, u(\boldsymbol{x}) \; w_{N\,|K}(\boldsymbol{x}) \, \mathrm{d}S = 0 \right\} \tag{3.4}$$

for all $w_N \in \mathcal{S}_0^{-1}(\mathcal{M})$. Here, $w_{N\,|K}$ means that the value of w_N on K has to used.

(3b) [I, 4 points]

If $\operatorname{supp}(u_0) \subset B_1 := \{\boldsymbol{x} \in \mathbb{R}^2 : \, \|\boldsymbol{x}\| < 1\}$, $\operatorname{div} \mathbf{a} = 0$, and final time $T = 1$, find the minimal $R > 0$ such that the (spatial) support of $u(\boldsymbol{x}, t)$ is contained in $]-R, R[^2$ for all $0 \le t \le T$.

HINT: Remember that $\|\mathbf{a}(\boldsymbol{x})\| \le 1$ for all $\boldsymbol{x}$.

For every pair (K, e), where K is a triangle of $\mathcal{M}$, and $e \in \mathcal{E}$ is one of the edges of K, we define the *upwind numerical flux* according to

$$F_e^K(v, w) := \begin{cases} \mathbf{a}(\boldsymbol{m}_e) \cdot \boldsymbol{n}_{\partial K}(\boldsymbol{m}_e) v & \text{, if } \mathbf{a}(\boldsymbol{m}_e) \cdot \boldsymbol{n}_{\partial K}(\boldsymbol{m}_e) \ge 0 \,, \\ \mathbf{a}(\boldsymbol{m}_e) \cdot \boldsymbol{n}_{\partial K}(\boldsymbol{m}_e) w & \text{, if } \mathbf{a}(\boldsymbol{m}_e) \cdot \boldsymbol{n}_{\partial K}(\boldsymbol{m}_e) < 0 \text{ and } e \not\subset \partial\Omega \,, \\ 0 & \text{, if } \mathbf{a}(\boldsymbol{m}_e) \cdot \boldsymbol{n}_{\partial K}(\boldsymbol{m}_e) < 0 \text{ and } e \subset \partial\Omega \,. \end{cases} \tag{3.5}$$

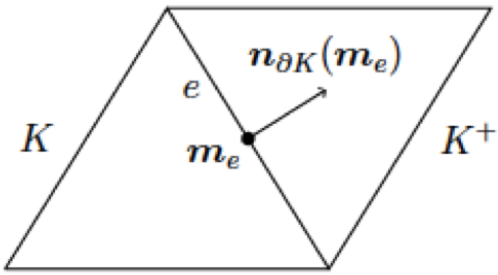

Figure 3.1: Illustration for numerical flux.

Here $\boldsymbol{m}_e$ is the midpoint of e and $\boldsymbol{n}_{\partial K}$ is the exterior unit normal for K, see Figure 3.1.

Then the upwind finite volume spatial semi-discretization of (3.1) on $\mathcal{M}$ boils down to the variational problem: seek $u_N(t) \in \mathcal{S}_0^{-1}(\mathcal{M})$ such that for all $w_N \in \mathcal{S}_0^{-1}(\mathcal{M})$

$$\sum_{K\in\mathcal{M}} \left[\int_K \frac{\partial u_N}{\partial t}(\boldsymbol{x},t) w_N(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} + \sum_{e\in\mathcal{E}, e\subset\partial K} |e| F_e^K(u_{N\,|K}\,, u_{N\,|K^+})\; w_{N\,|K} \right] = 0\;, \tag{3.6}$$

with $|e|$ the length of the edge, F_e^K from (3.5), and, given K and $e \subset \partial K$, K^+ a formal notation for the cell on the other side of e.

(3c) [I, 2 points]

At first glance (3.5) seems to be plagued by a failure to give a meaning to K^+ in case $e \subset \partial\Omega$. Explain why this need not worry us.

(3d) [I, 5 points]

Now we want to tackle the implementation of the finite volume discretization in LehrFEM. To that end implement a function

```
function nflux = uwfluxfn(Mesh,mu,Kidx,VHandle)
```

that returns a 3-vector of values $F_e^K(u_{N\,|K}\,, u_{N\,|K}^+)$ for all three edges of the mesh cell with number `Kidx`. The other arguments are

- `Mesh`: a LehrFEM mesh data structure complete with edge information, in particular the `Edge2Elem` field
- `mu`: the coefficient vector of u_N with respect to the global shape functions specified above (numbering as in (3.3)),
- `VHandle`: function handle of type `@(x)` supplying the velocity field $\mathbf{a}$ (returns a column vector).

You may use the LehrFEM function `getNormals(Mesh, i)` that returns a 2×3-matrix of exterior unit normals for triangle number i.

Using the basis of global shape functions as introduced above, (3.6) can be recast as an ordinary differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\vec{\boldsymbol{\mu}} = \mathbf{B}\vec{\boldsymbol{\mu}} \tag{3.7}$$

for the time-dependent coefficient vector of $u_M(\boldsymbol{x}, t)$. Here $\mathbf{B} \in \mathbb{R}^{N,N}$ is a sparse, fixed matrix.

(3e) [I, 2 points]

Which entries of $\mathbf{B}$ from (3.7) are guaranteed to be zero regardless of the choice of the velocity field $\mathbf{a}$?

(3f) [3 points]

Express the potential non-zero entries of $\mathbf{B}$ from (3.7) by means of the upwind numerical fluxes F_e^K from (3.5).

(3g) [8 points]

Write an *efficient* LehrFEM function

```
function B = assemble_Advec_FV(Mesh,VHandle)
```

that computes the matrix $\mathbf{B}$ from (3.7) for a given mesh passed as extended LehrFEM mesh data structure (with complete edge information) in `Mesh` and a velocity field provided by the function handle `VHandle`.

HINT: You may use the function `uwfluxfn` from sub-problem (3d) in the spirit of sub-problem (3f).

(3h) [I, 4 points]

Implement a LehrFEM function

```
mufinal = solve_Advec_FV(Mesh,VHandle,U0Handle,M)
```

that solves the initial boundary value problem (3.1) by means of the upwind finite volume method and the mesh passed as extended LehrFEM mesh data structure `Mesh` and returns an approximation for $u(\cdot, 1)$. The function handle arguments `VHandle` and `U0Handle` supply $\mathbf{a}$ and u_0 as functions defined on the domain covered by the mesh.

For timestepping use M equidistant timesteps of the the second-order Heun method (explicit midpoint rule, [NPDE, Equation 8.4.6]), described by the Butcher scheme

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \hline & 0 & 1 \end{array} . \tag{3.8}$$

For the implementation you may rely on the LehrFEM function

```
mu0 = samplecenters(Mesh,U0Handle)
```

that returns a vector of point values of u_0 at the centers of mesh cells.

HINT: Use the function `assemble_Advec_FV` from sub-problem (3k).

(3i) [2 points] We want to use the function `solve_Advec_FV` from the previous sub-problem for a sequence of meshes obtained by successive regular refinement. Why is it necessary to double the number of timesteps each time we proceed to a finer mesh?

(3j) [I, 5 points]

For testing the method we choose $\Omega =]-3,3[$, and

$$\mathbf{a}(\boldsymbol{x}) = \begin{pmatrix} -x_2 \\ x_1 \end{pmatrix} \quad , \quad u_0(\boldsymbol{x}) = \begin{cases} \cos^2(\pi \left\| \boldsymbol{x} - \binom{1}{0} \right\|) & \text{, if } \left\| \boldsymbol{x} - \binom{1}{0} \right\| \leq \frac{1}{2} \text{ ,} \\ 0 & \text{elsewhere.} \end{cases} \tag{3.9}$$

The functions are implemented in the MATLAB functions `vfield` and `u0function`. In addition, a MATLAB function `uexact` is given for the exact solution at time $t = 1$.

We use the upwind finite volume scheme discussed above on a sequence of meshes obtained by the global regular refinement of an initial coarse mesh of Ω. We track the discrete error norm

$$\|u(\cdot,1) - u_N(\cdot,1)\|_h^2 := \sum_{K \in \mathcal{M}} |K| |(u - u_N)(\boldsymbol{c}_K)|^2 \text{ ,}$$

where $\boldsymbol{c}_K$ is the barycenter of K, and $u_N(\cdot,1) \in \mathcal{S}_0^{-1}(\mathcal{M})$ is the approximate solution at final time $T = 1$ produced by the upwind finite volume method on the mesh $\mathcal{M}$ combined with equidistant timestepping with the explicit trapezoidal rule as implemented in `solve_Advec_FV`.

This norm can be computed by the supplied function `error = errornorm(Mesh, mua, mub)`, where `mua` and `mub` are vectors containing the values of the two functions u and u_N at the cell centers.

Augment the incomplete implementation of the MATLAB function `test_Adv_FV` so that it creates the doubly logarithmic ("loglog") plot of $\|u(\cdot,1) - u_N(\cdot,1)\|_h$ versus the mesh-width displayed in Figure 3.2.

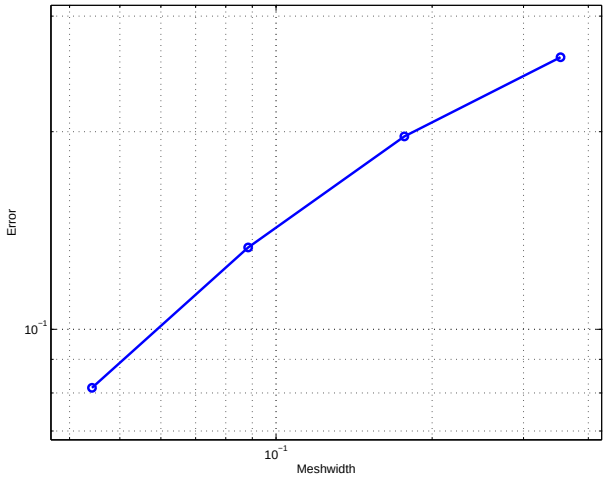

Figure 3.2: Error $\|u(\cdot,1) - u_N(\cdot,1)\|_h$ as a function of mesh-width h

(3k) [I, 2 points]

Describe in qualitative and quantitative terms the convergence of the method as observed in Figure 3.2.

HINT: The exact numbers can be found in Table 3.1.

h	$\|u - u_N\|_h$
0.35355	0.25976
0.17678	0.19673
0.088388	0.13328
0.044194	0.081441

Table 3.1: Meshwidths and errors from Figure 3.2.

Problem 4 Stable Evaluation at a Point [42 points]

On a bounded domain Ω with polygonal connected boundary $\partial\Omega$ we consider the Dirichlet problem for the Laplace equation

$$\Delta u = 0 \quad \text{in } \Omega \quad , \quad u = g \quad \text{on } \partial\Omega \; , \tag{4.1}$$

for a sufficiently smooth continuous function $g : \partial\Omega \mapsto \mathbb{R}$.

From now on let $\boldsymbol{x} \in \Omega$ be fixed. We are interested in an approximation of $u(\boldsymbol{x})$, that is, we aim for an approximate evaluation of the point value output functional

$$J(u) := u(\boldsymbol{x}) \; . \tag{4.2}$$

However, it is known that J is not continuous with respect to the energy norm associated with (4.1), which denies us the fast convergence endowed by duality arguments.

As a starting point for a suitable modification of J we consider the function

$$G_{\boldsymbol{x}}(\boldsymbol{y}) = -\frac{1}{2\pi} \log(\|\boldsymbol{x} - \boldsymbol{y}\|) \; , \quad \boldsymbol{x} \neq \boldsymbol{y} \; , \tag{4.3}$$

and the two functionals on $C^0(\overline{\Omega})$

$$P_{\mathrm{SL}}(v) := \int_{\partial\Omega} v(\boldsymbol{y}) G_{\boldsymbol{x}}(\boldsymbol{y}) \, \mathrm{d}S(\boldsymbol{y}) \; , \tag{4.4}$$

$$P_{\mathrm{DL}}(v) := \int_{\partial\Omega} v(\boldsymbol{y}) (\mathbf{grad}\, G_{\boldsymbol{x}})(\boldsymbol{y}) \cdot \boldsymbol{n}(\boldsymbol{y}) \, \mathrm{d}S(\boldsymbol{y}) \; , \tag{4.5}$$

where $\boldsymbol{n}$ is the exterior unit normal vector field on $\partial\Omega$.

The LehrFEM functions

```
function PSLval = PSL(Mesh,vhandle,x)
function PDLval = PDL(Mesh,vhandle,x)
```

evaluate P_{SL} and P_{DL}, respectively, for a triangulated polygonal domain. The parameter `Mesh` passes an extended LehrFEM mesh data structure with complete edge information. It also describes the domain Ω. The function handle `vhandle` of type `@(x)` gives the argument function v, and `x` is a 2-vector with the coordinates of $\boldsymbol{x}$.

(4a) [I, 8 points]

Implement the function

```
function PSLval = PSL(Mesh,vhandle,x)
```

The approximate evaluation of the integral should be done by means of the local midpoint rule on the partitioning of $\partial\Omega$ induced by the mesh. For the implementation you can use the already supplied function `function Gval = G(x,y)` that returns $G_{\boldsymbol{x}}(\boldsymbol{y})$; the input `x` is a *row* vector of coordinates and `y` has to be a *row* vector or a matrix where each line contains the coordinates of a point.

(4b) [I, 4 points]

Somebody contends that for a harmonic function u on Ω, that is, $\Delta u = 0$, holds

$$u(\boldsymbol{x}) = P_{\mathrm{SL}}(\mathbf{grad}\, u \cdot \boldsymbol{n}) - P_{\mathrm{DL}}(u)\ . \tag{4.6}$$

Complete the MATLAB script template `point_eval.m` that provides evidence for this claim for $\Omega =]0,1[^2$, $\boldsymbol{x} = \binom{0.3}{0.4}$, and $u(\boldsymbol{x}) = \log(\left\|\boldsymbol{x} + \binom{1}{0}\right\|)$, which satisfies $\Delta u = 0$ in Ω. To do so you should use the functions `PSL` and `PDL` for this u to evaluate the right hand side of (4.6) approximately for a sequence of meshes obtained by global regular refinement. Then you should plot the difference of both sides versus the mesh-width in a doubly logarithmic ('loglog') plot.

HINT: For the implementation use the supplied functions

```
uval = u(x,varargin)
gradunval = gradun(x)
```

to evaluate respectively u and $\mathbf{grad}\, u \cdot \boldsymbol{n}$ in the point $\boldsymbol{x}$.

The function `PDL` is available through the scrambled MATLAB file `PDL.p`.

(4c) [I, 4 points]

Assume that a function $\Psi \in C^2(\overline{\Omega})$ is given with the following properties:

(P1) $0 \le \Psi \le 1$;

(P2) $\Psi \equiv 1$ close to $\partial\Omega$;

(P3) $\Psi \equiv 0$ in a ball around $\boldsymbol{x}$.

By applying Green's formula [NPDE, Thm. 2.4.7], show that, if $\Delta u = 0$ in Ω,

$$P_{\mathrm{SL}}(\mathbf{grad}\, u \cdot \boldsymbol{n}) = \int_{\Omega} \mathbf{grad}\, u(\boldsymbol{y}) \cdot \mathbf{grad}(G_{\boldsymbol{x}}\Psi)(\boldsymbol{y})\, \mathrm{d}\boldsymbol{y}\ . \tag{4.7}$$

Note that $G_{\boldsymbol{x}}\Psi$ is a smooth function.

(4d) [I, 6 points]

Appealing to the result (4.7) of sub-problem (4c), use Green's formula [NPDE, Thm. 2.4.7] again to establish

$$\begin{aligned} P_{\mathrm{SL}}(\mathbf{grad}\, u \cdot \boldsymbol{n}) = &- \int_{\Omega} u(\boldsymbol{y})\ (2\,\mathbf{grad}\, G_{\boldsymbol{x}}(\boldsymbol{y}) \cdot \mathbf{grad}\, \Psi(\boldsymbol{y}) + G_{\boldsymbol{x}}(\boldsymbol{y})\Delta\Psi(\boldsymbol{y}))\ \mathrm{d}\boldsymbol{y} \\ &+ P_{\mathrm{DL}}(u)\ . \end{aligned} \tag{4.8}$$

HINT: Recall the formula $\Delta(vw) = v\Delta w + 2\,\mathbf{grad}\, w \cdot \mathbf{grad}\, v + w\Delta v$ and use that $\Delta G_{\boldsymbol{x}}(\boldsymbol{y}) = 0$ for $\boldsymbol{y} \neq \boldsymbol{x}$.

(4e) [I, 4 points]

From (4.8) and (4.6) we conclude that for the solution u of (4.1) we have the equivalent form of the point evaluation functional

$$J(u) = J^*(u) := -\int_{\Omega} u(\boldsymbol{y})\ (2\,\mathbf{grad}\, G_{\boldsymbol{x}}(\boldsymbol{y}) \cdot \mathbf{grad}\, \Psi(\boldsymbol{y}) + G_{\boldsymbol{x}}(\boldsymbol{y})\Delta\Psi(\boldsymbol{y}))\ \mathrm{d}\boldsymbol{y}\ . \tag{4.9}$$

Show that J^* is continuous on $H^1_0(\Omega)$ with respect to the energy norm associated with (4.1).

(4f) [I, 8 points] We consider $\Omega =]0,1[^2$ and know a priori that $\left\|\boldsymbol{x} - \binom{0.5}{0.5}\right\| \leq \frac{1}{4}$. Hence, independently of $\boldsymbol{x}$ we can choose

$$\Psi(\boldsymbol{y}) = \begin{cases} 0 & \text{, if } \left\|\boldsymbol{y} - \binom{0.5}{0.5}\right\| \leq \frac{1}{4}\sqrt{2}\ , \\ \cos^2\left(\frac{\pi}{\frac{1}{2}\sqrt{2}-1}\left(\left\|\boldsymbol{y} - \binom{0.5}{0.5}\right\| - \frac{1}{2}\right)\right)\ , & \\ 1 & \text{, if } \left\|\boldsymbol{y} - \binom{0.5}{0.5}\right\| \geq \frac{1}{2}\ . \end{cases} \tag{4.10}$$

which is in $C^2_{\mathrm{pw}}(\Omega)$.
This function is already implemented as

```
function [valPsi,gradPsi,laplPsi] = Psi(y) ,
```

which returns $\Psi(\boldsymbol{y})$, $\mathbf{grad}\, \Psi(\boldsymbol{y})$, and $\Delta\Psi(\boldsymbol{y})$.

Write a LehrFEM function

```
function val = Jstar(Mesh,mu,x)
```

that takes a LehrFEM mesh data structure for a triangular mesh of the square domain $\Omega =]0,1[^2$, the coefficient vector `mu` for a finite element function $u_N \in \mathcal{S}^0_1(\mathcal{M})$ (w.r.t to the standard nodal basis, of course), and the point coordinates `x` of $\boldsymbol{x} \in \Omega$ as arguments and returns an approximation of $J^*(u_N)$ from (4.9). Use the local quadrature rule P7O6() from the LehrFEM library for the approximate evaluation of the integral in (4.9).

HINT: For the implementation you can use the two already supplied functions `function Gval = G(x,y)` and `function Ggradval = Ggrad(x,y)` that return $G_{\boldsymbol{x}}(\boldsymbol{y})$ and $\mathbf{grad}\, G_{\boldsymbol{x}}(\boldsymbol{y})$ respectively.

(4g) [I, 4 points]

The supplied LehrFEM function `function mu = Solve_LaplDir_LFE(M,GHandle)` solves (4.1) by means of piecewise linear Lagrangian finite elements on a triangular mesh passed in the basic LehrFEM mesh data structure `Mesh`. The function handle `GHandle` gives the Dirichlet data g. The finite element solution is returned in the form of its coefficient vector `mu`.

Use this function together with `Jstar` implemented in sub-problem (4f) to realize a MATLAB function

```
function ux = stab_point_eval(Mesh,GHandle,x)
```

that uses the formula (4.8) together with a $\mathcal{S}_1^0(\mathcal{M})$-Galerkin finite element solution of (4.1) on $\Omega =]0,1[^2$ to compute $u(\boldsymbol{x})$. The point $\boldsymbol{x}$ must satisfy the assumptions of sub-problem (4f) and its coordinates are passed in `x`.

(4h) [I, 4 points]

For $\Omega =]0,1[^2$, $\boldsymbol{x} = \binom{0.3}{0.4}$, and $u(\boldsymbol{x}) = \log(\left\|\boldsymbol{x} + \binom{1}{0}\right\|)$ examine numerically the convergence of the value returned by `stab_point_eval` to $u(\boldsymbol{x})$. To that end perform the evaluation on a sequence of four meshes created by uniform regular refinement and plot the error on a doubly logarithmic plot ('loglog'). Describe the convergence in a qualitatively and quantitatively way.

Implement a MATLAB script achieving this by completing the template in `test_peval_cvg.m`. For the handle function for the Dirichlet boundary condition on you, remember that $g = u|_{\partial\Omega}$ on $\partial\Omega$.

HINT: Use `stab_point_eval` from (4g).

References

[NPDE] Lecture Slides for the course "Numerical Methods for Partial Differential Equations", SVN revision # 56087.

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

[LehrFEM] LehrFEM manual.