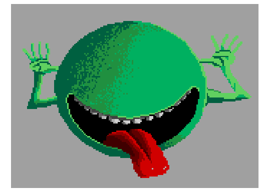
Course 401-0674-00L: Numerical Methods for Partial Differential Equations
Examination, 22.01.2013

Prof. Ralf Hiptmair



Don't panic !
Good luck !

Duration of examination: 180 minutes

The total number of points is 250, the maximum grade can be achieved with 170, passing requires 85. Please pay attention to the number of points awarded for each (sub-)task. It is roughly correlated with the amount of information your answer should contain. For additional information see the examination instruction sheet.

Problem 1. Finite volume method for scalar conservation law [70 points]

We consider the Cauchy problem on $\mathbb{R}\times]0,T[$ for the scalar conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\sin(\pi u) = 0 \;. \tag{1}$$

with initial condition $u(x,0) = u_0(x)$, $x \in \mathbb{R}$, satisfying

$$0 \leq u_0(x) \leq 1 \;, \quad x \in \mathbb{R} \;. \tag{2}$$

WARNING: The flux function $\sin(\pi u)$ is **concave** on $[0,1]$. Most of the examples in the lecture slides are for **convex** flux functions!

(1a) [5 points] Determine the entropy solutions of the Riemann problem with

$$\text{(i)} \quad u_0(x) = \begin{cases} 0 & \text{for } x \leq 0 \;, \\ 1 & \text{for } x > 0 \end{cases} \quad , \quad \text{(ii)} \quad u_0(x) = \begin{cases} 1 & \text{for } x \leq 0 \;, \\ 0 & \text{for } x > 0 \;, \end{cases}$$

(1b) [5 points] Assume $\mathrm{supp}(u_0) \subset [0,1]$ and (2). Describe the maximal support of the solution u of (1) in the x-t-plane.

(1c) [10 points] Write a MATLAB function

```
function flux = sinegodflux(v,w)
```

that implements the Godunov numerical flux $F_{\mathrm{GD}}(u,v) = f(u^{\downarrow}(v,w))$ for (1) under the assumption $0 \leq v, w \leq 1$.

HINT: Note that the flux function $f(u) = \sin(\pi u)$ is concave on $[0,1]$.

HINT: For reference a scrambled version of `sinegodflux` is available in `sinegodflux_ref.p`. It is vector-safe in the sense that you can pass vectors `v` and `w` and the result will be a vector of values $F_{\mathrm{GD}}(v,w)$.

(1d) [5 points] For the spatial semi-discretization of (1) we use a finite volume method on an equidistant spatial mesh covering the interval $[-6,6]$. We rely on the Godunov flux and assume that $u=0$ outside this interval for all relevant times.

Devise a MATLAB function

```
function rhs = sineclawrhs(mu)
```

that realizes the right hand side of the ODE, namely the expression

$$-\frac{1}{h}\left(F(\mu_j,\mu_{j+1})-F(\mu_{j-1},\mu_j)\right)$$

arising from this finite volume semi-discretization of (1) on a mesh of $[-6,6]$ with N equal cells, taking an N-vector `mu` with cell averages.

Here, h is the mesh size and F is the Godunov numerical flux function.

HINT: An implementation of `sineclawrhs` is provided in `sineclawrhs_ref.p`. It assumes `mu` is a column vector.

(1e) [5 points] For timestepping we use the explicit trapezoidal rule, a 2nd-order explicit Runge-Kutta single step method characterized by the Butcher tableau

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \quad . \tag{3}$$

Implement a MATLAB function

```
function yend = expltrpz(g,y0,T,M)
```

that integrates the abstract ODE $\dot{\mathbf{y}}=\mathbf{g}(\mathbf{y})$ over $[0,T]$ with initial state vector $\mathbf{y}_0$ using M equidistant timesteps of the explicit trapezoidal rule (3). The `g`-argument passes a handle of type `@(y)` to the function g.

HINT: The file `expltrpz_ref.p` provides a reference implementation.

(1f) [10 points] Relying on `sineclawrhs` and `expltrpz` write a MATLAB script `solvesineclaw.m` that solves the Cauchy problem for (1) numerically over the time interval $[0,1]$ with

$$u_0(x)=\begin{cases}1 & \text{for } 0\le x<1\ , \\ 0 & \text{elsewhere.}\end{cases}$$

Sample the initial data at midpoints of mesh cells. Use a spatial mesh with $N=600$ cells and $M=200$ timesteps and plot the numerical solution for $T=1$. Save your plot in the file `sineclaw.eps`.

HINT: Do not forget axis labels!

(1g) [10 points] Assuming a uniform spatial meshwidth $h>0$ determine the constraint on the timestep such that the numerical domain of dependence is guranteed to cover the exact domain of dependence (CFL-conditon) for the method used in the previous sub-problem.

(1h) **[10 points]** Write a MATLAB script `findtimestep` that determines the maximum possible timestep size for the numerical experiment of sub-problem (1f).

HINT: You may employ a bisection type approach along with examining blow-up of solutions.

(1i) **[5 points]** Copy `sineclawrhs.m` with your implementation of `sineclawrhs` to `sineclawrhs_reac.m`. In this file extend that function so that it can deal with the augmented conservation law with reaction term

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\sin(\pi u) = -u \; . \tag{4}$$

HINT: Recall the finite volume interpretation of the coefficients of the vector argument `mu`.

(1j) **[5 points]** Conduct the numerical experiment of sub-problem (1f) for (4) and save the plot in the file `sineclaw_reac.eps`.

Problem 2. Trace error estimates [55 points]

On a convex polygon $\Omega \subset \mathbb{R}^2$ with exterior unit normal $\boldsymbol{n}$ we consider the boundary value problem

$$-\Delta u = f \quad \text{in } \Omega \quad , \quad \boldsymbol{n} \cdot \mathbf{grad}\, u + u = 0 \quad \text{on } \partial\Omega \; , \tag{5}$$

with source function $f \in L^2(\Omega)$.

(2a) **[5 points]** State the variational formulation of (5).

HINT: Do not forget the appropriate function spaces.

(2b) **[5 points]** Discuss existence and uniqueness of solutions of the variational formulation obtained in sub-problem (2a).

(2c) **[10 points]** The boundary value problem (5) is discretized by means of linear Lagrangian finite elements on a sequence $(\mathcal{M}_i)$ of triangular meshes of Ω created by uniform regular refinement of a coarse mesh $\mathcal{M}_0$. Let $u_i \in \mathcal{S}_1^0(\mathcal{M}_i)$ denote the finite element solution on mesh $\mathcal{M}_i$.

Quantitatively, predict the asymptotic behavior of the error norms $|u - u_i|_{H^1(\Omega)}$ and $\|u - u_i\|_{L^2(\Omega)}$ in terms of $N_i := \dim \mathcal{S}_1^0(\mathcal{M}_i)$.

(2d) **[10 points]** In the setting of the previous sub-problem, what asymptotic behavior can be expected from the error norm $\|u - u_{N,i}\|_{L^2(\partial\Omega)}$ on the boundary in terms of $N_i := \dim \mathcal{S}_1^0(\mathcal{M}_i)$.

HINT: Use the multiplicative trace inequality.

(2e) **[10 points]** The LehrFEM function `solveImpedanceBVP(mesh)` (provided in the file `solveImpedanceBVP.m`) solves (5) for $f \equiv \cos(\|\boldsymbol{x}\|)$ on a polygon using linear Lagrangian finite elements on triangular meshes. The domain is specified through the argument `mesh` that passes a basic LehrFEM mesh data structure as returned from `load_Mesh`.

Extend this code such that it returns the quantity $B(u_N) := \int_{\partial\Omega} u_N \,\mathrm{d}S$, where u_N is the finite element solution.

(2f) **[10 points]** Measure the rate of convergence of $B(u_N) \to B(u)$ by computing the numerical values on meshes obtained by successive refinement of the triangular mesh (of a regular hexagon) stored in `Coord_Hexagon.dat` and `Elem_Hexagon.dat`. Do this in a MATLAB script `convergence.m`.

HINT: The "exact" value is $B(u) = 2.08154105279$.

HINT: To refine a mesh, use the LehrFEM function `refine_REG`.

(2g) **[5 points]** Prove that

$$B(u) = \int_\Omega f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \;. \tag{6}$$

HINT: Use the variational formulation.

Problem 3. Mehrstellenverfahren for Poisson equation [70 points]

We consider the scalar second-order boundary value problem

$$-\Delta u = f \quad \text{in } \Omega :=]0,1[^2 \quad , \quad u = 0 \quad \text{on } \partial\Omega \;, \tag{7}$$

for a bounded and continuous function $f \in C^0(\overline{\Omega})$.

The so-called *Collatz Mehrstellenverfahren* can be viewed as a modified finite element Galerkin discretization using bilinear finite elements on a uniform quadrilateral tensor product mesh of Ω with meshwidth $h := (M+1)^{-1}$, $M \in \mathbb{N}$. We take for granted that standard nodal basis functions are used.

Then the Mehrstellenverfahren is obtained when using the following 4×4 element matrix on each cell K of the mesh

$$\mathbf{A}_K := \frac{1}{6}\begin{pmatrix} 5 & -2 & -1 & -2 \\ -2 & 5 & -2 & -1 \\ -1 & -2 & 5 & -2 \\ -2 & -1 & -2 & 5 \end{pmatrix} \;, \tag{8}$$

and the following element vector

$$\vec{\boldsymbol{\varphi}}_K = \frac{h^2}{12}\begin{pmatrix} 2f_1 + f_2 + f_4 \\ 2f_2 + f_3 + f_1 \\ 2f_3 + f_4 + f_2 \\ 2f_4 + f_1 + f_3 \end{pmatrix} \quad , \quad f_i := f(\boldsymbol{a}_i) \;, \quad i = 1,2,3,4 \;. \tag{9}$$

Here, we have denoted the vertices of the square cell K by $\boldsymbol{a}_i$, $i = 1,2,3,4$, and have used the counter-clockwise local numbering indicated in Figure 1.

Throughout this problem we assume a lexikographic numbering of the interior vertices of the mesh and of the associated basis functions, see Figure 2.

Eventually, we arrive at a linear system of equations $\mathbf{A}\vec{\boldsymbol{\mu}} = \vec{\boldsymbol{\varphi}}$ encoded by (8) and (9).

(3a) **[5 points]** What is the size of the matrix $\mathbf{A}$ and of the right hand side vector $\vec{\boldsymbol{\varphi}}$?

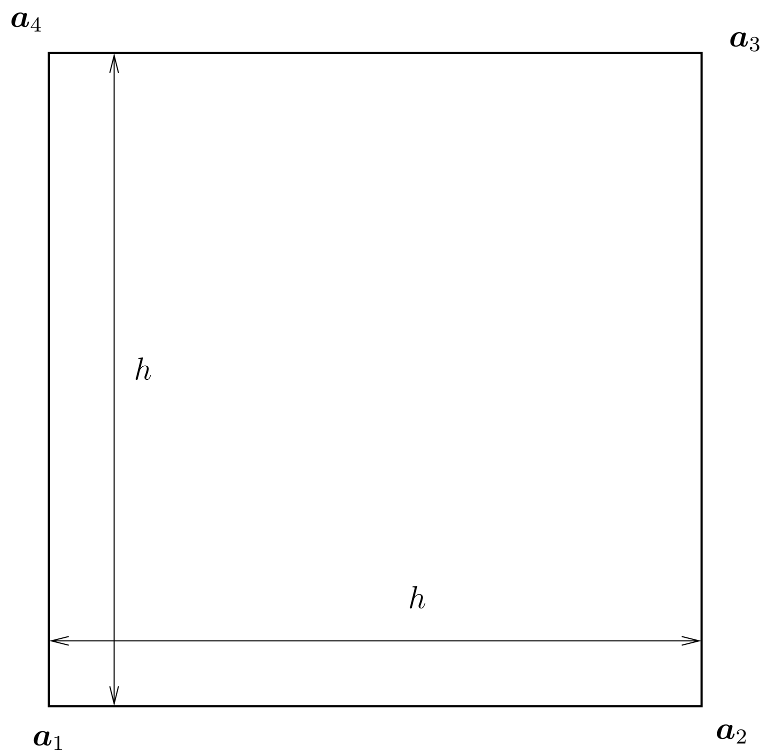


Figure 1: Local numbering of vertices and associated local shape functions for a square cell K.

(3b) **[5 points]** What is the meaning of the entries of the solution vector $\vec{\boldsymbol{\mu}}$?

(3c) **[10 points]** Write a MATLAB function

```
function A = compMehrstellenA(M)
```

that assembles the *sparse* Galerkin matrix of the Mehrstellenverfahren.

HINT: The matrix $\mathbf{A}$ is a block tridiagonal matrix with tridiagonal blocks.

HINT: In MATLAB `A = gallery('tridiag',n,c,d,e)`, where `c`, `d`, and `e` are all scalars, yields the Toeplitz tridiagonal matrix of order `n` with subdiagonal elements `c`, diagonal elements `d`, and superdiagonal elements `e`.

HINT: The MATLAB command `kron` that computes the Kronecker product of two matrices

$$\mathbf{R} \otimes \mathbf{Q} = \begin{pmatrix} r_{1,1}\mathbf{Q} & r_{1,2}\mathbf{Q} & \cdots & r_{1,M}\mathbf{Q} \\ r_{2,1}\mathbf{Q} & r_{2,2}\mathbf{Q} & \cdots & r_{2,M}\mathbf{Q} \\ \vdots & & \ddots & \vdots \\ r_{M,1}\mathbf{Q} & \cdots & \cdots & r_{M,M}\mathbf{Q} \end{pmatrix} \in \mathbb{R}^{MN,MN} \ , \quad \mathbf{R} \in \mathbb{R}^{M,M}, \ \mathbf{Q} \in \mathbb{R}^{N,N} \ ,$$

comes handy.

(3d) **[15 points]** Devise a MATLAB function

```
function f = compMehrstellenf(f,M)
```

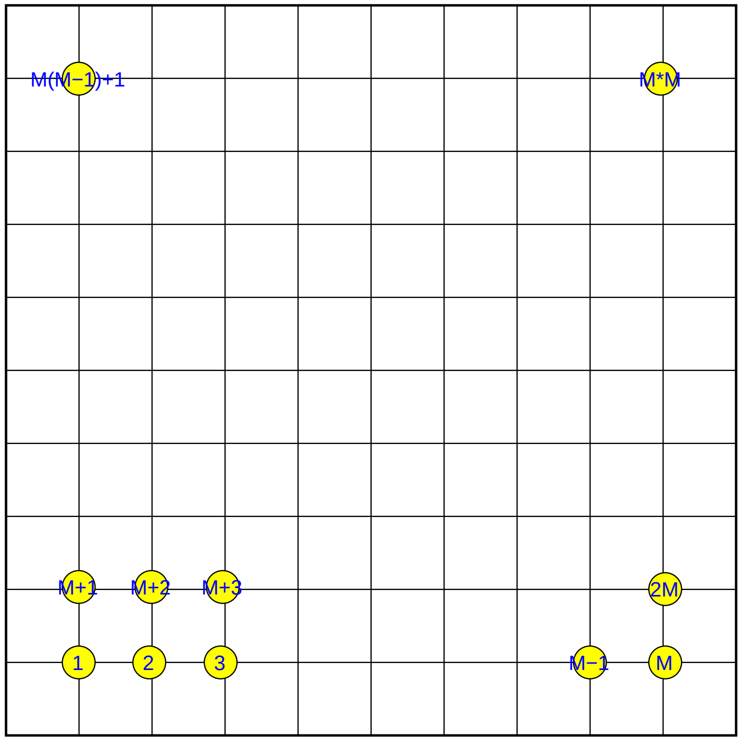


Figure 2: Lexikographic numbering of vertices of the equidistant tensor product mesh

that returns the right hand side vector $\vec{\varphi}$ for the Mehrstellenverfahren. The argument `f` passes a handle of type `@(x)` (with a 2-vector `x`) to the function f, while $M+1$ gives the number of grid cells in one direction.

(3e) **[5 points]** Write a MATLAB function

```
function u = solveMehrstellen(f,M)
```

that computes the coefficient vector $\vec{\boldsymbol{\mu}}$ for the Mehrstellen discretization of (7), when supplied with a handle `f` to the source function f and the discretization parameter `M` (see sub-problem (3e)).

(3f) **[10 points]** We consider $f(x,y) = \sin(\pi x)\sin(\pi y)$, which yields the exact solution $u(x,y) = (2\pi^2)^{-1} f(x,y)$, $(x,y) \in \Omega$.

Write a MATLAB function

```
function err = compgriderr(M)
```

that computes the discrete norm of the discretization error

$$\|u - u_M\|_{\mathrm{mst}} := \max\{|u(\boldsymbol{x}) - u_M(\boldsymbol{x})| : \ \boldsymbol{x} = (ih, jh), \ 1 \leq i,j \leq M, \ h := (M+1)^{-1}\} \ . \tag{10}$$

where u_M is the "Mehrstellen solution" for discretization parameter `M` defined at the vertices of the mesh.

HINT: A reference implementation of `solveMehrstellen` is supplied in the file `solveMehrstellen_ref.p`.

(3g) **[10 points]** Write a MATLAB script `convergence.m` that estimates the (algebraic) order of convergence of the Mehrstellenverfahren with respect to the error norm (10) and for the specific source function $f(x,y) = \sin(\pi x)\sin(\pi y)$. To achieve this, evaluate the error for $M = 5, 10, 20, 40, 80, 160$ using the function `compgriderr` from sub-problem (3f).

HINT: A reference implementation of `compgriderr` can be accessed through the file `compgriderr_ref.p`.

(3h) **[10 points]** Show that for non-negative f, the Mehrstellen solution of (7) cannot have negative values at the vertices of the mesh.

HINT: Use an indirect argument (proof by contradiction) as in the proof of the discrete maximum principle given in the lecture.

Problem 4. Stabilized Galerkin method for convection-diffusion in 1D [55 points]

We consider the one-dimensional stationary advection-diffusion boundary value problem

$$-\epsilon \frac{\mathrm{d}^2 u}{\mathrm{d}x^2} + \frac{\mathrm{d}u}{\mathrm{d}x} = f \quad \text{in} \quad \Omega :=]0,1[\quad , \quad u(0) = u(1) = 0 \; , \tag{11}$$

for some source function $f \in L^2(\Omega)$ and $\epsilon > 0$.

The so-called *non-symmetric weak penalty formulation* of (11) amounts to the following linear variational problem: seek $u \in C^1_{\mathrm{pw}}([0,1])$ such that

$$\mathsf{a}(u,v) + \mathsf{b}(u,v) + \mathsf{c}(u,v) = \int_0^1 f v \,\mathrm{d}x \quad \forall v \in C^1_{\mathrm{pw}}([0,1]) \; , \tag{12}$$

with

$$\mathsf{a}(u,v) := \int_0^1 \epsilon \frac{\mathrm{d}u}{\mathrm{d}x}\frac{\mathrm{d}v}{\mathrm{d}x}\,\mathrm{d}x \quad , \quad \mathsf{b}(u,v) := \int_0^1 \frac{\mathrm{d}u}{\mathrm{d}x} v \,\mathrm{d}x \; ,$$

$$\mathsf{c}(u,v) := -\epsilon\left(\frac{\mathrm{d}u}{\mathrm{d}x}(1)v(1) - \frac{\mathrm{d}u}{\mathrm{d}x}(0)v(0) - u(1)\frac{\mathrm{d}v}{\mathrm{d}x}(1) + u(0)\frac{\mathrm{d}v}{\mathrm{d}x}(0)\right) + \sigma u(1)v(1) \; ,$$

where $\sigma > 0$ is a penalty parameter at our disposal.

In this problem we study the Galerkin finite element discretization of (12) on equidistant meshes

$$\mathcal{M} := \{]ih,(i+1)h[,\ i = 0,\dots,N-1\} \; , \quad h := \frac{1}{N} \; , \quad N \in \mathbb{N} \; ,$$

based on the trial spaces of continuous, piecewise linear finite element functions on $\mathcal{M}$ that vanish in $x = 0$:

$$V_N := \{V_N \in \mathcal{S}^0_1(\mathcal{M}) :\ v_N(0) = 0\} \; . \tag{13}$$

The composite trapezoidal rule on $\mathcal{M}$ is applied for the approximate evaluation of the right hand side with continuous f.

Throughout, the standard “tent function” nodal basis of V_N will be used. The basis functions are assumed to be numbered lexikographically “from left to right”.

(4a) **[5 points]** Show that a (smooth) solution of (11) solves (12).

(4b) **[10 points]** Determine the element matrices $\mathbf{A}_K$, $\mathbf{B}_K$ and $\mathbf{C}_K$ associated with the bilinear forms a, b and c for

(i) a generic interior mesh cell $K =]ih, (i+1)h[$, $i = 1, \ldots, N-2$,

(ii) the rightmost mesh cell $K_0 =]1-h, 1[$.

(4c) **[10 points]** Write a MATLAB function

```
function G = compGalMatab(epsilon,N)
```

that computes the Galerkin matrix for the bilinear form $\mathsf{a} + \mathsf{b}$ on an equidistant mesh with `N` cells. As trial space use V_N from (13) equipped with the nodal basis.

HINT: The MATLAB command `A = gallery('tridiag',c,d,e)` returns the tridiagonal matrix with subdiagonal `c`, diagonal `d`, and superdiagonal `e`, which can be scalars or vectors.

HINT: A scrambled reference implementation of `compGalMatab` is provided in the file `compGalMatab_ref.p`

(4d) **[5 points]** Write a MATLAB function

```
function A = compGalMat(epsilon,sigma,N)
```

that returns the finite element Galerkin matrix for the bilinear form from (12). As trial space use V_N from (13). The arguments pass ϵ, the penalty parameter σ, and the number N of mesh cells.

HINT: Use the function `compGalMatab` from sub-problem (4c).

(4e) **[10 points]** Write a MATLAB function

```
funtion rhs = comprhs(f,N)
```

that computes the right hand side vector for the Galerkin discretization of (12) described above. The argument `N` passes the number of cells of the mesh, whereas `f` is a handle of type `@(x)` to the right hand side function `f`.

HINT: Recall that the composite trapezoidal rule should be used.

(4f) **[5 points]** Write a MATLAB script `solve1DCDBVP(epsilon,sigma,N)` that plots the finite element Galerkin solution of (11) for $f \equiv 1$. The arguments provide ϵ, the penalty parameter σ for the variational formulation (12), and the number N of mesh cells.

Create the plot for $\epsilon = 10^{-2}$, $\sigma = 10\epsilon N$, $N = 300$, and store it in the file `stabgal.eps`.

(4g) **[5 points]** We write $u_N \in V_N$ for the Galerkin solution of (12) with $f \equiv 1$, $\epsilon = 10^{-2}$ computed on a mesh with N cells and penalty parameter $\sigma = 10\epsilon N$.

Table 1 provides the $L^2(\Omega)$-norms of discretization errors for different N. Using these data, describe the observed L^2-convergence of the approximate solutions u_N qualitatively and quantitatively.

N	40	60	80	160	320	640	1280
$\|u - u_N\|_{L^2(\Omega)}$	0.024	0.012	0.0072	0.0019	0.00050	0.00013	0.000032

Table 1: (Approximate) $L^2(\Omega)$-norm of the discretization error of the finite element Galerkin discretization of (12)

HINT: The data from the table are stored in the file `stabgerr.dat` and can be loaded into an 2×8 matrix `stabgerr` via `load('stabgerr.dat')`.

(4h) **[5 points]** Solve sub-problem (4f) for a finite element Galerkin discretization based on the trial and test spaces

$$V_{N,0} := \mathcal{S}_1^0(\mathcal{M}) \cap H_0^1(\Omega) \ .$$

To that end copy your MATLAB script `solve1DCDBVP.m` to `solve1DCDBVPclass.m` and modify the latter file.

HINT: What is the dimension of $V_{N,0}$? The modifications affect only a single row and column of the Galerkin matrix and a single entry of the right hand side vector.

Problem References

[NPDE] Lecture Slides for the course "Numerical Methods for Partial Differential Equations", SVN revision # 54024.

Last modified on April 11, 2013