R. Hiptmair
L. Scarabosio
C. Urzua Torres

Spring Term 2015

Numerical Methods for Partial Differential Equations

ETH Zürich
D-MATH

Homework Problem Sheet 10

Problem 10.1 Dirichlet BVP with Delta-Function Right-Hand Side (Core problem)

One of the messages of [NPDE, Section 2.4.2] was that not all linear variational problems on function spaces are well-posed, see [NPDE, Def. 2.4.12]. In this problem we will encounter a specimen of an innocent looking ill-posed linear variational problem that was discussed in detail in [NPDE, Ex. 2.4.18]. We are going to study how the ill-posed nature of the continuous variational problem will be reflected by the behavior of Galerkin solutions. This will be done empirically based on a `C++` implementation using DUNE.

Template files for the new classes you will need to write are available in the lecture `svn` repository

`assignments_codes/assignment10/Problem1`

The idea is that you extend your own code, reason why it does not contain the files you already implemented in the previous assignments. Please do not forget to include them when you submit your work. If your own implementation is still not working, you may use the classes provided in `assignments_codes/solutions`.

In this problem we consider the linear variational problem

$$u \in H^1_0(\Omega): \quad \int_\Omega \operatorname{grad} u(\boldsymbol{x}) \cdot \operatorname{grad} v(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = v(0) \quad \forall v \in H^1_0(\Omega), \tag{10.1.1}$$

posed on a polygon Ω (with $0 \in \Omega$) and attempt its numerical solution by means of a linear finite element Galerkin discretization.

(10.1a) Describe a physical system for which (10.1.1) provides a model of certain aspects. What is the meaning of u in this model.

HINT: Remember a particular model discussed in [NPDE, Chapter 2].

(10.1b) Which problem haunts the linear variational problem (10.1.1)? What can you say about the existence of a minimizer of the associated quadratic functional?

HINT: Refresh yourself on the contents of [NPDE, Section 2.2.3]. Again study the beginning on [NPDE, Section 2.3] and [NPDE, Ex. 2.4.18].

(10.1c) Let $\mathcal{M}$ be a triangular mesh of Ω. The Galerkin discretization of (10.1.1) based on $\mathcal{S}^0_{1,0}(\mathcal{M})$ leads to a discrete variational problem, *cf.* [NPDE, Section 3.2]. Explain why the associated discrete quadratic minimization problem (see [NPDE, Eq. (1.5.6)]) always has a unique solution.

(10.1d) Let $\mathcal{M}_k$ be a sequence of triangular meshes, where $\mathcal{M}_{k+1}$ is generated from $\mathcal{M}_k$ by regular refinement of all triangles. Guess how the minimal value of the quadratic functional associated with the variational problem (10.1.1) will behave as $k \to \infty$.

(10.1e) Implement the class LocalDelta which provides a method

```
template <class Element>
  void operator()(Element const& e, ElementVector &local) const;
```

to compute the element vector associated to the specific right-hand side of (10.1.1). The method should check whether 0 belongs to the given element and then compute the corresponding values using linear Lagrangian finite elements.

HINT: Note this is similar to LocalFunction implemented in subproblem (7.4e).

(10.1f) We want to equip our finite element code with the capability to computer the $L^2(\Omega)$- and $H^1(\Omega)$-seminorm of piecewise linear finite element functions given through their coefficient vectors with respect to the nodal basis of $\mathcal{S}_1^0(\mathcal{M})$.

For this reason, now you are asked to complete the class `Norms` by implementing the methods methods L2Norm, and H1sNorm, whose names already tell their function. They both take as argument a coefficient vector `Q`.

```
template <class DofHandler>
class Norms{
public:
  using calc_t = double;
  using GridView = typename DofHandler::GridView;
  enum{ world_dim = GridView::dimension };
  using MatrixType  = Eigen::SparseMatrix<calc_t,Eigen::RowMajor>;
  using Coordinate  = Dune::FieldVector<calc_t, world_dim>;

  Norms(DofHandler const& dof_handler)
    : dofh_(dof_handler), gv_(dofh_.gv), N_(dofh_.size()) {};

  template <class Vector>
  double L2Norm(Vector const& Q);

  template <class Vector>
  double H1sNorm(Vector const& Q);

private:
  DofHandler dofh_;
  unsigned N_;
  GridView const& gv_;
};
```

HINT: Remember that both norms are the "energy norms" induced by suitable bilinear forms. Thus the Galerkin matrices for these forms can be used to compute the (squares of the) norms.

(10.1g) Now we consider (10.1.1) on the unit disk $\Omega := \{\boldsymbol{x} \in \mathbb{R}^2 : \|\boldsymbol{x}\| < 1\}$. An incomplete file `main.cc` is supplied. The loading and refinement of the mesh, as well as the computation

of the stiffness matrix is already implemented.

Extend this code to compute the load vector using the function you wrote in (10.1a), solve the system and obtain the L^2-norm and H^1-seminorm of the solution. Finally plot the L^2-norm and H^1-norm of the solutions vs. the number of degrees of freedom, and also output the solution for the finest grid in `Vtk`-format.

(10.1h) Consider an abstract linear variational problem

$$u \in V_0 : \quad \mathsf{a}(u,v) = \ell(v) \quad \forall v \in V_0, \tag{10.1.2}$$

with s.p.d. ($\rightarrow$ Def. [NPDE, Def. 2.2.35]) bilinear form a and a linear form ℓ that is continuous in the energy norm in the sense of [NPDE, Eq. (2.2.48)].

Elaborate the relationship between the energy norm of the solution of (10.1.2) and the minimal value of the associated quadratic functional according to [NPDE, § 2.4.2].

(10.1i) Explain your observations on the behavior of the H^1-norm in subproblem (10.1g) in light of the theoretical conclusions obtained in subproblems (10.1d) and (10.1g).

Problem 10.2 Debugging Finite Element Codes (Core problem)

[NPDE, Chapter 5] confronted you with theoretical results on the asymptotic convergence of finite element Galerkin solutions for 2nd-order elliptic boundary value problems. On the one hand, these estimates can be used to gauge the relative efficiency of different finite element approximations, as was discussed in [NPDE, Section 5.3.5]. On the other hand, expected rates of convergence are a fine probe for detecting errors in a finite element code. For instance, the observed convergence to a known analytic solution should match the theoretical predictions, unless the code is flawed. Another way of using the approximation results of [NPDE, Section 5.3.5] to identify a faulty finite element implementation is demonstrated in this problem.

In detail these considerations are presented in [NPDE, Section 5.8]. This problem complements this section of the lecture material, in particular [NPDE, § 5.8.9]. Study this paragraph again before you continue.

The required files for this problem are available in the lecture `svn` repository

`assignments_codes/assignment10/Problem2`

Three different local assemblers

$$\mathsf{LocalLaplaceQFEX}, \quad \mathtt{X} \in \{1,2,3\}$$

purport to provide the Galerkin matrix and right-hand side vector for the finite element discretization of the variational problem

$$u \in H^1(\Omega) : \quad \mathsf{a}(u,v) := \int_\Omega \operatorname{grad} u \cdot \operatorname{grad} v \, \mathrm{d}\boldsymbol{x} = \ell(v) := \int_\Omega f(\boldsymbol{x}) v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \quad \forall v \in H^1(\Omega) \tag{10.2.1}$$

using *quadratic* Lagrangian finite elements (space $\mathcal{S}_2^0(\mathcal{M})$) on a triagular mesh $\mathcal{M}$ of some polygon $\Omega \subset \mathbb{R}^2$. The local assemblers provide the same basic methods as the `LocalAssembler` object in [NPDE, Code 3.6.49]. The implied ordering of local shape functions is the same as in [NPDE, Ex. 3.6.41], see also [NPDE, Code 3.6.42].

(10.2a) Complete the class

```
template <class DofHandler>
  class InterpolateQFE{
  public:
    using calc_t = double;
    using GridView = typename DofHandler::GridView;
    enum { K=2 };
    enum { world_dim = GridView::dimension };

    InterpolateQFE(DofHandler const& dof_handler) :
       dofh_(dof_handler), gv_(dofh_.gv) {};

    template <class Vector, class Function>
    void operator()(Vector &Phi, Function const& f) const;

  private:
    DofHandler const& dofh_;
    GridView const& gv_;
  };
```

by writintg operator() that accepts a function f in procedural form, and fills Phi with the basis coefficients of the nodal interpolant $\mathsf{I}_2 u \in \mathcal{S}_2^0(\mathcal{M})$. This particular piecewise quadratic interpolation is presented in [NPDE, Ex. 3.5.3].

(10.2b) Determine a sharp bound $T(h_{\mathcal{M}})$ in the estimate

$$|\mathsf{a}(u,u) - \mathsf{a}(\mathsf{I}_2 u, \mathsf{I}_2 u)| \leq C T(h_{\mathcal{M}}), \tag{10.2.2}$$

where $u : \overline{\Omega} \mapsto \mathbb{R}$ is supposed to be smooth and the unknown constant $C > 0$ may depend only on Ω and the shape regularity measure of $\mathcal{M}$.

Use the following result, which is a generalization of [NPDE, Cor. 5.3.43] to quadratic Lagrangian finite element spaces.

Theorem. *Let $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, be a bounded polygonal/polyhedral domain equipped with a simplicial mesh $\mathcal{M}$. Then the following interpolation error estimate holds for the nodal interpolation operator I_2 onto $\mathcal{S}_2^0(\mathcal{M})$*

$$\|u - \mathsf{I}_2 u\|_{H^1(\Omega)} \leq C h^{\min\{3,k\}-1} |u|_{H^k(\Omega)} \quad \forall u \in H^k(\Omega), \quad k = 2, 3\,,$$

with a constant $C > 0$ depending only on k and the shape regularity measure $\rho_{\mathcal{M}}$.

HINT: Recall from [NPDE, Section 5.1] the arguments that suggested that the energy norm provides a relevant norm for measuring the discretization error.

(10.2c) Write a method

```
template <class LocalAssembler>
double test_assembleQFE(DofHandler const & dofh,
                        LocalAssembler lassemblr)
```

that returns $\mathsf{a}(\mathsf{l}_2 u, \mathsf{l}_2 u)$ for $u(\boldsymbol{x}) = \exp(\|\boldsymbol{x}\|^2)$ and the domain triangulated by the mesh described by the `Dune::GridView` and the finite element space managed by DofHandler dofh. The argument lassemblr passes a local assembler for quadratic Lagrangian finite element with the calling syntax of LocalLaplaceQFEX introduced above.

HINT: Use InterpolateQFE developed in (10.2a).

(10.2d) Complete the file `main.cc` so for each refinement level and for each of the local assemblers LocalLaplaceQFEX, $\mathtt{X} \in \{1, 2, 3\}$, it computes $errX = |\mathsf{a}(u, u) - \mathsf{a}(\mathsf{l}_2 u, \mathsf{l}_2 u)|$ for the function $u(\boldsymbol{x}) = \exp(\|\boldsymbol{x}\|^2)$ from (10.2c). Finally plot $errX$ against the number of elements in a suitable scale.

HINT: Use the method test_assembleQFE implemented in (10.2c).

Also, you may use $|u|^2_{H^1(\Omega)} = 23.7608$ for $\Omega = (0, 1)^2$.

(10.2e) Which implementations of the assembly routine are wrong, which are correct? Explain your answer.

Problem 10.3 Crouzeix-Raviart Finite Elements

In this problem we come across an alien "non-conforming" piecewise linear finite element space. It has no direct relevance for scalar second-order elliptic boundary value problems, but permits us to practice notions and techniques for finite elements.

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^j_N, $j = 1, \ldots, N$, which satisfy

$$b^i_{N\,|K} \in \mathcal{P}_1(K) \quad \forall K \in \mathcal{M} \quad , \quad b^i_N(\boldsymbol{m}_j) = \begin{cases} 1 & \text{, if } i = j \text{ ,} \\ 0 & \text{else,} \end{cases} \quad i, j \in \{1, \ldots, N\} \text{ .} \tag{10.3.1}$$

(10.3a) Show that (10.3.1) provides a valid definition of the functions b^j_N.

(10.3b) Show that the set of functions $\{b^j_N : j = 1, \ldots, N\}$ is linearly independent.

(10.3c) Show that $\mathcal{CR}(\mathcal{M}) \not\subset H^1(\Omega)$.

(10.3d) Describe the support of a basis function b^i_N.

(10.3e) We use the b^i_N from (10.3.1) 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:

$$b^j_{N\,|K} = 1 - 2\lambda_{\mathrm{opp}(j)} \text{ ,} \quad j = 1 \ldots, N \text{ ,} \tag{10.3.2}$$

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

(10.3f) 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 Dune we use the b_N^j from (10.3.1) as global shape functions. We number them taking into account the numbering of edges in Dune, see [NPDE, § 3.6.21].

As usual, template files for the new classes you will need to write are available in the lecture `svn` repository

`assignments_codes/assignment10/Problem3`

The idea is that you extend your own code, reason why it does not contain the files you already implemented in the previous assignments. Please do not forget to include them when you submit your work. If your own implementation is still not working, you may use the classes provided in `assignments_codes/solutions`.

(10.3g) Implement the class CRDofHandler as you did for DofHandler but considering now the edges (co-dimension 1) as degrees of freedom instead of the vertices (co-dimension 2).

HINT: This means you have to change **operator**() and size()

(10.3h) We introduce the mesh-dependent bilinear form

$$\mathsf{a}_{\mathcal{M}}(u,v) := \sum_{K\in\mathcal{M}} \int_K \mathbf{grad}\, u \cdot \mathbf{grad}\, v \,\mathrm{d}\boldsymbol{x} \;. \tag{10.3.3}$$

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, Fig. 85]. Follow the convention that the i-th local edge is opposite to the i-th local vertex, $i = 1,2,3$.

HINT: Use (10.3.2) and [NPDE, Eq. (3.3.21)].

(10.3i) Implement the class CRLocalLaplace which provides a method

```
template <class Element, class Matrix>
void operator()(Element const& e, Matrix &local) const;
```

that computes the element matrix for $\mathsf{a}_{\mathcal{M}}$ using the Crouzeix-Raviart finite element space.

HINT: The local numbering scheme of the previous sub-problem does not apply, use the Dune intrinsic numbering convention ([NPDE, Rem. 3.6.24]).

HINT: Recall the result of the subproblem (10.3h) and use AnalyticalLocalLaplace from subproblem (7.4b) that computes the element matrices for $\mathsf{a}_{\mathcal{M}}$ for the piecewise linear Lagrangian finite element spaces.

(10.3j) Prove that the Galerkin matrices for $\mathsf{a}_{\mathcal{M}}$ and the finite element spaces $\mathcal{CR}(\mathcal{M})$ are always positive semidefinite.

(10.3k) 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 (10.3.1) is one-dimensional and spanned by the vector with all entries $= 1$.

(10.3l) Implement a method

```
template <class Function>
double L2Norm(DofHandler const& dofh, Vector const& Q, Function
    const& u)
```

in `main.cc` that takes the DofHandler dofh for a particular GridView, a coefficient vector `Q` of length N describing a function $u_N \in \mathcal{CR}(\mathcal{M})$ (in `Q`), and a Function u in procedural form, representing $u : \Omega \mapsto \mathbb{R}$ (in `u`). The return value should provide an approximation for $\|u - u_N\|_{L^2(\Omega)}$ computed by means of the local numerical quadrature offered by Dune with order 10.

HINT: Notice we want to compute the $\|u - u_N\|_{L^2(\Omega)}$, where u is a continuos function and u_N a finite element approximation function, i.e., $u_N(x) = \sum_{k=0}^{M} Q_j b_N^j(x)$. Therefore, in each element you need to use the local basis functions to reconstruct $u(x)_N$ and then use quadrature integrate $|u(x) - u_N(x)|^2$. You may follow the structure of the alternative solution for the H1 semi-norm in (8.2k). If you do, don't forget to consider the basis functions instead of the gradients.

(10.3m) 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

$$u_N(\boldsymbol{m}) = g(\boldsymbol{m}) \quad \forall \boldsymbol{m} \in \mathcal{N}_\partial \quad , \quad \mathsf{a}_{\mathcal{M}}(u_N, v_N) = 0 \quad \forall v_N \in \mathcal{CR}_0(\mathcal{M}) \; . \tag{10.3.4}$$

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 method `void solve(DofHandler const & dof, Vector & uN)` in `main.cc` which:

- Marks the boundary Dofs and set them inactive
- 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. The choice of global shape functions and numbering schemes as introduced above still apply.
- Assembles the right hand side vector.
- Solves the system, i.e. computes the coefficient vector (w.r.t. the basis $\{b_N^j\}_{j=1}^N$ from (10.3.1)) for the solution $u_N \in \mathcal{CR}(\mathcal{M})$ of (10.3.4). The obtained coefficient vector for u_N should have N components, where N is the number of *all* edges in the mesh $\mathcal{M}$.

HINT: You may rely on the function CRLocalLaplace from subproblem (10.3i) and CRBoundaryDofs in the repository.

(10.3n) Somebody claims that the Crouzeix-Raviart finite element space and, in particular, the method solve() from sub-problem (10.3m) can be used to solve the boundary value problem

$$-\Delta u = 0 \quad \text{in } \Omega, \quad u = g \quad \text{on } \partial\Omega\,, \tag{10.3.5}$$

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.

To that end complete the implementation of `main.cc` so it otputs a vector of the values $\|u - u_N\|_{L^2(\Omega)}$ and a vector of the number of elements for each mesh.

(10.3o) 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 return values you should get can be loaded from the MATLAB data file `l2err_CR.dat`.

Published on 29.04.2015.
To be submitted on 06.05.2015.

References

[NPDE] Lecture Slides for the course "Numerical Methods for Partial Differential Equations".SVN revision # 75265.

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

Last modified on April 30, 2015