Institut für Numerische Simulation
Rheinische Friedrich-Wilhelms-Universität Bonn

universitätbonn

Computer lab Numerical Algorithms
Winter term 2012/2013
Prof. Dr. M. Rumpf – B. Geihe, B. Heeren

Problem sheet 1 — October 23rd, 2012

We consider the cell problem posed in the lecture: For a given (linear) profile u_H we want to compute a periodic correction u_h such that $-\mathrm{div}\,(a(u_H + u_h)) = 0$ and $\fint u_h \,\mathrm{d}x = 0$. Using the periodicity assumptions the weak formulation is:

$$\int_Q a(x)\, \nabla u_h(x) \cdot \nabla \varphi_h(x)\,\mathrm{d}x = -\int_Q a(x)\, \nabla u_H(x) \cdot \nabla \varphi_h(x)\,\mathrm{d}x \quad \forall\, \varphi_h \in \mathcal{V}_h$$

For discretization we consider a triangulation $\mathcal{T}_h$ and a corresponding set of finite element basis functions $\{\varphi_h^i\}_{i\in I}$. By writing $u_h = \sum_{j\in I} u_h^j \varphi_h^j$ and choosing basis functions as test functions in the weak formulation above we derive a linear system of equations $A\bar{u}_h = \bar{b}$ where

$$A_{ij} = a\left(\varphi_h^i, \varphi_h^j\right) = \int_Q a(x)\, \nabla\varphi_h^i(x) \cdot \nabla\varphi_h^j(x)\,\mathrm{d}x, \qquad b_i = (A\bar{u}_H)_i$$

The periodic boundary conditions will be enforced by modifying the stiffness matrix A and the right hand side $\bar{b}$.

We will start by assembling the stiffness matrix A which is usually done in the following way:

Assemblation of the stiffness matrix A

---

foreach *triangle* T **do**
    foreach *pair of basis functions* φ_i, φ_j *which have* T *in their support* **do**
        foreach *quadrature point* γ_T *with weight* ω_{γ_T} **do**
            $A_{ij} \mathrel{+}= |T| * \omega_{\gamma_T} * a(T, \gamma_T, i, j)$

An implementation of this procedure can be seen in class `LinearFEOperator`. The function $a(T, \gamma_T, i, j)$ must now evaluate the integrand $a(x(\gamma_T))\, \nabla\varphi_h^i(x(\gamma_T)) \cdot \nabla\varphi_h^j(x(\gamma_T))$ at the given quadrature point γ_T in T for basis functions φ_h^i and φ_h^j. It will be implemented in a separate class `IsoDiffusiveBilf` .

This class must be able to evaluate the diffusivity a at a given point. Therefore it has a member variable `TensorOrder0`. To allow for more general diffusivity terms later the class `TensorOrder0` is pure virtual and just defines that every derived class must supply the required evaluate method. We will implement a derived class BlobTensor which realizes the following diffusivity:

$$a(x) = \psi_\varepsilon\left(\|x - m\|\right), \quad \text{where } \psi_\varepsilon(s) = -\frac{1}{\pi}\arctan\left(\frac{2s-1}{\varepsilon}\right) + \frac{1}{2}$$

The midpoint m and the "transition width" $\varepsilon \in (0.01, 0.5)$ should be selectable.

To take periodic boundary conditions into account we propose to first assemble the usual stiffness matrix A above and then perform modifications on A to reflect periodicity. This will be done by class PeriodicBoundaryMask derived from BoundaryMask (as is DirichletBoundaryMask which was used in problem 1).
In the constructor of this class a mapping needs to be generated which maps boundary nodes to their periodic counterparts. This could e. g. be realized using a vector of pairs of integers: std :: vector< SmallVector2<**int**> >. Here one has to decide which nodes get eliminated by the periodic identification. Moreover the class should provide a method apply which collapses the matrix as discussed on problem sheet 1. To tackle the right hand side accordingly a method collapse should be implemented. Finally method extend is supposed to copy values at remaining boundary nodes to their eliminated counterparts.

So far the additional constraint $\fint u_h \, \mathrm{d}x = 0$ has not been accounted for. In the discrete setting the non uniqueness of solutions to the periodic cell problem will result in a rank deficiency of the resulting stiffness matrix. While in practice a CG method started from a feasible point $\bar{u}_h$ might just iterate in the correct linear subspace we want to enforce this by additional projection steps. Therefore the constraint is discretized in the known manner:

$$0 = \fint u_h \, \mathrm{d}x = |Q|^{-1} \int_Q u_h \, 1 \, \mathrm{d}x = |Q|^{-1} \sum_{i,j} u_h^i \int_Q \varphi_i \varphi_j \, \mathrm{d}x$$

$$\rightsquigarrow |Q|^{-1} \, \bar{u}_h^\top \, M \, \bar{1} = 0 \quad \text{where } M_{ij} = \int_Q \varphi_i \varphi_j \, \mathrm{d}x$$

That way $\tilde{n} := |Q|^{-1} M \, \bar{1}$ is orthogonal to every $\bar{u}_h$ fulfilling the constraint. With a normalized normal vector $n := \tilde{n} / \|\tilde{n}\|$ the orthogonal projection of an arbitrary point x is given by $P(x) = x - (x \cdot n) \, n$. This projection may be performed after each step of a CG method as it does not affect the residual. To set up the normal vector n a mass matrix M can be generated analogously to the stiffness matrix by using the bilinear form MassBilf.

Tasks:

- Implement the scalar tensor BlobTensor derived from TensorOrder0
- Complete the class IsoDiffusiveStiffBilf
- For the class PeriodicBoundaryMask implement the constructor and methods apply, collapse and extend
- In the main function setup the right hand side from the given Vector u_H and perform the necessary periodic identification operations
- Implement projectingCGapply to perform a usual CG iteration with additional orthogonal projection along a supplied Vector constr ($= n$)