Exercise Sheet 3

07.05.2021 – 14.05.2021

Computational Fluid Dynamics (Summer Term 2021)

Exercise 1 (Programming). Nonlinear Poisson equation (15 Points)
In this exercise, you learn how to solve nonlinear PDEs and how to implement mixed Dirichlet-Neumann boundary conditions. To this end, we consider the nonlinear Poisson equation

$$\begin{aligned} -\nabla \cdot ((1+\kappa u^2)\nabla u) &= f \quad \text{on } \Omega = [0,1]\times[0,1] \\ u &= g_D \text{ on } \Gamma_D := \{0\}\times[0,1] \\ (1+\kappa u^2)\nabla u \cdot \vec{n} &= g_N \text{ on } \Gamma_N := \partial\Omega \setminus \Gamma_D \end{aligned}$$

with $f(x,y) = 1$ and boundary values

$$\begin{aligned} g_D(x,y) &= \begin{cases} 0 & y \leq 0.5 \\ -1 & y > 0.5 \end{cases} \\ g_N(x,y) &= 1 \end{aligned}$$

The corresponding nonlinear variational formulation is given by:

find $u \in u_D + V$ with $V := H^1_{\Gamma_D}(\Omega)$, $u_D \in H^1(\Omega), u_{|_{\Gamma_D}} = g_D$ such that

$$F_\kappa(u;v) := \int_\Omega (1+\kappa u^2)\nabla u \cdot \nabla v - fv \mathrm{d}x - \int_{\Gamma_N} g_N v \mathrm{d}\sigma = 0 \text{ for all } v \in V. \tag{1}$$

Nonlinear equations of this type can be solved by using Newton's method:

a. Choose initial iterate $u^{(0)} \in u_D + V$

b. for $k = 0, \ldots$:
find $\delta u \in V$ with

$$D_u F_\kappa(u^{(l)}; \delta u, v) = F_\kappa(u^{(l)}; v) \text{ for all } v \in V. \tag{2}$$

compute new iterate $u^{(l+1)} := u^{(l)} - \delta u$

(2) is a linear PDE with searched solution δu. Here, the bilinear form $D_u F_\kappa(u^{(l)}; \cdot, \cdot)\colon V \times V \to \mathbb{R}$ denotes the Jacobian of F_κ w.r.t. argument u at point $u^{(l)}$:

$$D_u F_\kappa(u^{(l)}; \delta u, v) = \int_\Omega (1+\kappa(u^{(l)})^2)\nabla\delta u \cdot \nabla v + 2u^{(l)}\delta u \nabla u^{(l)} \cdot \nabla v \mathrm{d}x.$$

HiFlow provides an implementation of Newton's method for solving nonlinear problems of type (1). In order to use this method, the user has to provide assembly routines for the bilinear form $D_u F_\kappa(u^{(l)}; \cdot, \cdot)$ (left-hand side of (2)) and the linear form $F_\kappa(u^{(l)}; \cdot)$ (right-hand side of (2)).

a. Modify the matrix assembly routine of the class LocalPoissonAssembler at the location marked by TODO exercise A in ex2_NonlinearPoisson.h to compute $D_u F(u^{(l)}; \cdot, \cdot)$.

Hint: In this routine, the data fields sol_ns_[q], grad_sol_ns_[q] store the values of the previous Newton iterate at quadrature point q, i.e. $u^{(l)}(x_q)$ and $\nabla u^{(l)}(x_q)$, respectively.

b. Modify the vector assembly routine of the class LocalPoissonAssembler at the location marked by TODO exercise B in ex2_NonlinearPoisson.h to compute the volume integral part $\int_\Omega$ of $F(u^{(l)}; \cdot)$.

c. Modify the vector assembly routine of the class LocalPoissonAssembler at the location marked by TODO exercise C in ex2_NonlinearPoisson.h to compute the surface integral part $\int_{\Gamma_N}$ of $F(u^{(l)}; \cdot)$.

Hint: Use the material number of the current facet e to check whether $e \subset \Gamma_N$ or $e \subset \Gamma_D$.

d. Modify the evaluate(face, pt_coord, vals) routine of the struct DirichletBC at the location marked by TODO exercise D in ex2_NonlinearPoisson.h to evaluate the Dirichlet function g_D.

Hint: If the provided object face e is not part of the Dirchlet boundary Γ_D, then the return object vals should be an empty vector.

e. Run the code on 1 or 2 mpi processes for $\kappa = 1$, visualize the results ex2_solution4.pvtu - ex2_solution7.pvtu with paraview, apply the filter warp by scalar to u and submit the corresponding screenshots.

Note that the parameter κ is defined in the parameter file ex2_poisson.xml and can be changed without recompilation.

f. How does the magnitude of κ affect the convergence speed of Newton's method and why could this be the case?

g. The concept of *numerical continuation* can be used to enhance the convergence of nonlinear solvers. This method is given as follows:

(a) given initial value κ_0 such that the Newton iteration for $F_{\kappa_0}(u) = 0$ converges reasonably fast and a number of continuation steps n_c. This gives a solution u_0.

(b) for $i = 1, \ldots, n_c$:

   i. set $\kappa_i = \kappa_0 + \frac{i-1}{n_c-1}(\kappa - \kappa_0)$

   ii. call Newton's method for solving $F_{\kappa_i}(u) = 0$ with starting value u_{i-1}

   iii. the solution is denoted by u_i.

Implement this method at the location marked by TODO exercise G in ex2_NonlinearPoisson.cc (routine solve_system(initial_kappa, kappa, num_steps). Run the code for $\kappa_0 = 1$, $\kappa = 100$, $n_c = 10$ and state the number of Newton iterations that are needed for the final value κ.

The corresponding code framework is ex2_NonlinearPoisson. Don't forget to the modify the file exercises/CMakeLists.txt accordingly, see the first exercise sheet for details.

Submission: until 14.05.2021, 11 am, moodle upload